DFCv2

Louis Granboulan[1], Phong Q. Nguyen[1], Fabrice Noilhan[2] and Serge Vaudenay[3]

[1] École Normale Supérieure, Département d'Informatique
[2] Université d'Orsay, Laboratoire de Recherche en Informatique
[3] Swiss Federal Institute of Technology (EPFL)
`Louis.Granboulan@ens.fr`, `Phong.Nguyen@ens.fr`, `Fabrice.Noilhan@ens.fr`, `Serge.Vaudenay@epfl.ch`

Abstract The development process of the Advanced Encryption Standard (AES) was launched in 1997 by the US government through NIST. The Decorrelated Fast Cipher (DFC) was the CNRS proposal for the AES, among 14 other candidates in 1998. It was based on the recent decorrelation theory, to obtain certain security proofs covering linear and differential cryptanalysis. DFC received numerous comments. In particular, Coppersmith discovered a weakness in the key schedule. We address this weakness by a slight modification on DFC. This paper presents the specifications and rationales of DFC version 2, and discusses issues raised during the AES process.

1 Introduction

A major goal in cryptography is to prove security statements on encryption schemes. To this respect, it is well-known that the status of secret-key cryptography is quite different from that of public-key cryptography. The decorrelation theory was introduced in 1998 (see [20] for the original reference) as an attempt towards filling this gap, by providing new ideas to build block ciphers, together with security proofs covering certain (however general) classes of attacks. Since the AES process was launched by NIST at about the same period, the French National Center for Scientific Research (CNRS) decided to start a project aimed at showing that decorrelation theory was a reasonable proposal for making secure and efficient block ciphers. The target platform was chosen to be 64-bit microprocessors, as such chips are likely to become standard during the lifetime of the AES. The CNRS project gave birth to the "Decorrelated Fast Cipher" (DFC) [6,7].

Decorrelation theory (see [20,21,22,23,24,25]) enables to prove formal results on the security of cryptographic primitives under certain hypotheses which we believe to be realistic. In particular, it enables to quantify the best advantage to distinguish two families of block ciphers, for a class of attacks with limited resources. For instance, one can consider any Turing machine restricted to a given number d of oracle calls to the block cipher. Most of the existing block ciphers are provably secure for the $d = 1$ case. However, none addresses the

$d = 2$ case, except DFC and other decorrelation theory–based ones. Interestingly, the $d = 2$ case already provides formal security against possible formalizations of differential and linear cryptanalysis. The Nyberg-Knudsen approach [16] was the only previously known way to achieve similar security statements (with MISTY [13,14] as a famous example.) The MISTY approach however does not provide much design flexibility, and the DFC approach seems to achieve stronger results as shown in Section 4. Besides, the Nyberg-Knudsen approach is indeed an ad hoc construction for providing security against differential and linear attacks but does not consider other general attacks with $d = 2$.

Implementing decorrelated block ciphers with order $d = 2$ by using known techniques (like the PEANUT construction [20]) requires the use of built-in multiplication which leads to non-trivial optimization tricks. DFC was submitted to the AES in order to show that such challenges could be overcome. DFC attracted many comments from the AES community, sometimes controversial. For instance, it was claimed that DFC was too slow, that its security paradigm brought nothing new, and that the security margin was too small. In addition, Coppersmith discovered a weakness in the key schedule by showing the existence of a fraction of 2^{-128} of weak keys (using a quite complex algorithm).

In this paper, we give the complete specifications of DFCv2. This new version addresses the key schedule problem and allows scalable modifications of the internal structure (so that the user can choose any "security margin"). We also try to respond to the issues raised on the original DFC.

2 Specifications of DFCv2

In this section, we give the complete specifications of DFCv2, and emphasize rationales in each subsection. A sample test vector for the nominal choices of the parameters is given in Appendix.

2.1 Notation

All quantities are bit strings or integers. When string lengths are divisible by four, quantities are denoted in hexadecimal. For instance, $\mathtt{d43}_\mathrm{x}$ denotes the bitstring `110101000011` and also represents the (decimal) integer 3395 in arithmetic operations. We use classical bitwise bitstring operations: OR, AND, NOT, XOR. We also use the following arithmetic operations over the integers: $+$, $\times$, mod. The result of an arithmetic operation is implicitly converted into a bitstring whose length will be clear from the context. Finally, we use the bitstring concatenation $|$ and the trunc_n function that extracts the n leftmost bits of a bitstring.

2.2 High Level Overview

DFCv2 is characterized by four parameters m, k, r and s chosen for security and efficiency reasons. In DFCv2(m, k, r, s), m is the message block length, k is the key length, r is the number of encryption rounds, and s is the number of rounds for the subkey generation. We require that $m \geq 32$, $0 \leq k \leq 2m$, $rs \leq 128$, m is a multiple of 4, and r is even. The nominal choice for DFCv2 is $m = 128$, $k \in \{128, 192, 256\}$, $r = 8$ and $s = 4$.

The encryption function DFC_K operates on m-bit message blocks by means of a secret key K of arbitrary length k up to $2m$ bits. The corresponding decryption function is DFC_K^{-1} and operates on m-bit message blocks.

The secret key K is first turned into an mr-bit "Expanded Key" EK through an "Expanding Function" EF, *i.e.* $\mathrm{EK} = \mathrm{EF}(K)$. As explained in Section 2.5, the EF function applies r s-round Feistel schemes (see Feistel [5]). The encryption process itself performs a similar r-round Feistel scheme. Each round uses the "Round Function" RF. This function maps a $\frac{m}{2}$-bit string onto a $\frac{m}{2}$-bit string by using one m-bit string parameter. It is defined in Section 2.3.

Given a bitstring σ of length multiple of m, say $m\rho$, we split it into ρ m-bit strings

$$\sigma = p_1|p_2|\ldots|p_\rho.$$

From σ we define a permutation Enc_σ on the set of m-bit strings coming from an ρ-round Feistel scheme. For any m-bit string PT which is split into two $\frac{m}{2}$-bit halves x_0 and x_1 so that $\mathrm{PT} = x_0|x_1$. We build a sequence $x_0, \ldots, x_{\rho+1}$ by the equation

$$x_{i+1} = \mathrm{RF}_{p_i}(x_i) \text{ XOR } x_{i-1} \quad (i = 1, \ldots, \rho) \tag{1}$$

and we define $\mathrm{Enc}_\sigma(m) = x_{\rho+1}|x_\rho$.

Given an m-bit plaintext block PT and the mr-bit expanded key EK, the $\mathrm{DFCv2}_K$ encryption function is obtained as

$$\mathrm{DFCv2}_K = \mathrm{Enc}_{\mathrm{EK}} \tag{2}$$

(that is, an r-round Feistel Cipher).

The EF function uses an s-round version defined with Enc.

If we split EK into r m-bit strings

$$\mathrm{EK} = \mathrm{RK}_1|\mathrm{RK}_2|\ldots|\mathrm{RK}_r \tag{3}$$

obviously, we have $\mathrm{DFC}_K^{-1} = \mathrm{Enc}_{\mathrm{revEK}}$ where

$$\mathrm{revEK} = \mathrm{RK}_r|\mathrm{RK}_{r-1}|\ldots|\mathrm{RK}_1. \tag{4}$$

2.3 The RF Function

The RF function (as for "Round Function") is fed with one m-bit parameter, which we view as two $\frac{m}{2}$-bit parameters: an "a-parameter" and a "b-parameter". It processes a $\frac{m}{2}$-bit input x and outputs a $\frac{m}{2}$-bit string defined as follows:

$$\mathrm{RF}_{a|b}(x) = \mathrm{CP}\left(((a \times x + b) \bmod p) \bmod 2^{\frac{m}{2}}\right) \tag{5}$$

where CP is a permutation over the set of all $\frac{m}{2}$-bit strings (which appears in Section 2.4) and p is the smallest prime integer greater than $2^{\frac{m}{2}}$. For instance, if $m = 128$, we use $p = 2^{64} + 13$. See the following table for other values.

m	p
32	$2^{16}+1$
64	$2^{32}+15$
96	$2^{48}+21$
128	$2^{64}+13$

Following the PEANUT scheme paradigm (see [20]), the RF function implements a decorrelation module. It is basically made from a classical round function (with CP), and from the pairwise decorrelation module $x \mapsto (ax+b \bmod p) \bmod 2^{\frac{m}{2}}$ *which was used in the PEANUT construction.*

From this construction, Decorrelation Theory ensures that if we consider DFCv2$(128,k,6,s)$ *and if we make the heuristic assumption that EK is random and uniformly distributed from the random choice of the secret key, then the best advantage for distinguishing this reduced and idealized version of DFCv2 from a truly random permutation when limited to two chosen plaintexts is less than* 2^{-117} *(see [24]). This property has several consequences on the formal security of DFCv2 as summarized in Section 4.*

2.4 The CP Permutation

The CP permutation (as for "Confusion Permutation") uses a look-up table RT (as for "Round Table") which takes a 6-bit integer as input and provides a $\frac{m}{4}$-bit string output. Its size is thus $2m$ bytes.

Let $y = y_l|y_r$ be the input of CP where y_l and y_r are two $\frac{m}{4}$-bit strings. We define

$$\mathrm{CP}(y) = ((y_r \text{ XOR } (\mathrm{RT} \circ \mathrm{trunc}_6)(y_l))|(y_l \text{ XOR } \mathrm{KC})) + \mathrm{KD} \bmod 2^{\frac{m}{2}} \qquad (6)$$

where KC is a $\frac{m}{4}$-bit constant string, and KD is a $\frac{m}{2}$-bit constant string. The permutation CP is depicted in Fig. 1.

The constants $\mathrm{RT}(0), \ldots, \mathrm{RT}(63)$, KC and KD will be set in Section 2.6.

The purpose of CP is to implement a permutation over all $\frac{m}{2}$*-bit strings which breaks the algebraic structure of the decorrelation module. For this we use a mixture of XORs and additions in a way very similar to that of the RC5 block cipher [19].*

The RT tables play an important role by introducing randomness. These tables are limited to $2m$ *bytes in total (in order to fit to embedded hardware with low memory) but with a maximal input size.*

2.5 Key Scheduling Algorithm

In order to generate a sequence $\mathrm{RK}_1, \mathrm{RK}_2, \ldots, \mathrm{RK}_r$ from a given key K represented as a bit string of length at most $2m$, we use the following algorithm. We first pad K with a constant pattern KS in order to make a $2m$-bit "Padded Key" string by

$$\mathrm{PK} = \mathrm{trunc}_{2m}(K|\mathrm{KS}). \qquad (7)$$

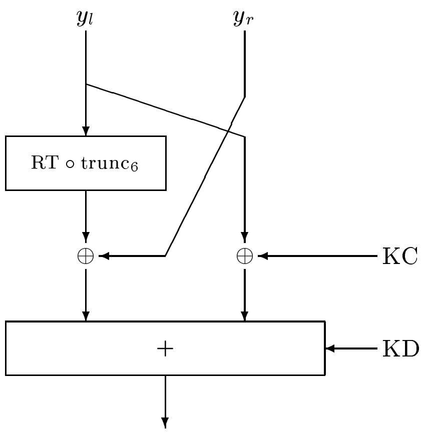

Figure1. The CP Permutation.

If K is of length m, we can observe that only the first m bits of KS are used. We define KS of length $2m$ in order to allow any key size from 0 to $2m$.

Then we split PK into two m-bit strings RK_0 and IRK_0 (as for "Internal Round Key") such that $\text{PK} = \text{IRK}_0|\text{RK}_0$.[1] We assume we are given 16 m-bit constants $\text{KAB}_0, \ldots, \text{KAB}_{15}$.[2] We now define

$$\text{IRK}_{j+1} = \text{IRK}_j \text{ XOR } \begin{cases} \text{KAB}_{\text{RT}(j) \bmod 16} & \text{if } j < 64 \\ \text{KAB}_{(\text{RT}(j-64)>>8) \bmod 16} & \text{otherwise} \end{cases} \tag{8}$$

for $j = 0, 1, \ldots, rs - 1$ where $\text{RT}(j-64) >> 8$ denotes the bitstring $\text{RT}(j-64)$ logically shifted by 8 bits to the right. Basically, we take the four least significant bits of $\text{RT}(j)$ for $j < 64$ and some other four bits of $\text{RT}(j-64)$ for $64 \leq j < 128$. (Since we require that $rs \leq 128$, j is less than 128.) We notice that IRK_j is actually the XOR of IRK_0 with some constant depending on j.

Each sequence of s IRK_j values defines an sm-bit string IEK_i which serves as the round key sequence of some s-round internal encryption function. More precisely, we define

$$\text{IEK}_i = \text{IRK}_{is-s+1}|\ldots|\text{IRK}_{is-1}|\text{IRK}_{is} \tag{9}$$

for $i = 1, 2, \ldots, r$ and

$$\text{IEnc}_i = \text{Enc}_{\text{IEK}_i} \tag{10}$$

for $i = 1, 2, \ldots, r$ as an "Internal Encryption". We now define the RK_i sequence by

$$\text{RK}_i = \text{IEnc}_i(\text{RK}_{i-1}) \tag{11}$$

---

[1] The following IRK_i sequence replaces the $\text{OAP}_i|\text{OBP}_i$ and $\text{EAP}_i|\text{EBP}_i$ sequences defined in DFCv1.

[2] These constants replace the KA_i and KB_i sequences defined in DFCv1.

for $i = 1, 2, \ldots, r$. Finally we define

$$\mathrm{EK} = \mathrm{EF}(K) = \mathrm{RK}_1|\mathrm{RK}_2|\ldots|\mathrm{RK}_r. \tag{12}$$

We can start the same process from $\mathrm{IRK}_r|\mathrm{RK}_r$ instead of PK. This enables to decrypt by computing the reversed sequence RK_i "on the fly".

This new key schedule repairs two drawbacks which were reported on DFCv1 (see [2]). Namely, due to the pairwise difference of the IRK_i*s, the iterations of the* IEK_i*s are no longer symmetric which fixes the weak key property reported by Coppersmith, and the first round key* RK_1 *now depends on all key bits. In addition, the* RK_i *sequence now looks "more random".*

2.6 On the Definition of the Constants

The previously defined algorithm depends on several constants:

- 64 constants $\mathrm{RT}(0), \ldots, \mathrm{RT}(63)$ of $\frac{m}{4}$ bits (thus forming $16m$ bits),
- one $\frac{m}{2}$-bit constant KD,
- one $\frac{m}{4}$-bit constant KC,
- 16 m-bit constants $\mathrm{KAB}_0, \ldots, \mathrm{KAB}_{15}$
- one $2m$-bit constant KS.

Those constants must satisfy the following security criterion.

1. the RT round table has no collision,
2. KD is odd,
3. the IRK_j are pairwise different for $j = 1, \ldots, rs$.[3]

We will use some constants several times. Actually, the RT table, KC and KD will contain the other constants. We thus need $18m$ bits of random constants.

In order to convince that this design hides no trap-door, we choose the constants from the hexadecimal expansion of the mathematical e constant

$$e = \sum_{n=0}^{\infty} \frac{1}{n!} = 2.\mathtt{b7e151628aed2a6abf7158}_{\mathrm{x}} \ldots \tag{13}$$

We use the following scheme in order to define the constants.

Step 1. Let EES (as for "e Expansion String") be the first $18m$ bits of the expansion of e after the (hexa)decimal point, we define

$$\mathrm{trunc}_{\frac{67}{4}m}(\mathrm{EES}) = \mathrm{RT}(0)|\mathrm{RT}(1)|\ldots|\mathrm{RT}(63)|\mathrm{KD}|\mathrm{KC}. \tag{14}$$

[3] Note that when this criterion is satisfied for one key, it is satisfied for any key.

Here is the EES string for $m = 128$.

$\texttt{b7e15162 8aed2a6a bf715880 9cf4f3c7 62e7160f 38b4da56}_{\mathrm{x}}$
$\texttt{a784d904 5190cfef 324e7738 926cfbe5 f4bf8d8d 8c31d763}_{\mathrm{x}}$
$\texttt{da06c80a bb1185eb 4f7c7b57 57f59584 90cfd47d 7c19bb42}_{\mathrm{x}}$
$\texttt{158d9554 f7b46bce d55c4d79 fd5f24d6 613c31c3 839a2ddf}_{\mathrm{x}}$
$\texttt{8a9a276b cfbfa1c8 77c56284 dab79cd4 c2b3293d 20e9e5ea}_{\mathrm{x}}$
$\texttt{f02ac60a cc93ed87 4422a52e cb238fee e5ab6add 835fd1a0}_{\mathrm{x}}$
$\texttt{753d0a8f 78e537d2 b95bb79d 8dcaec64 2c1e9f23 b829b5c2}_{\mathrm{x}}$
$\texttt{780bf387 37df8bb3 00d01334 a0d0bd86 45cbfa73 a6160ffe}_{\mathrm{x}}$
$\texttt{393c48cb bbca060f 0ff8ec6d 31beb5cc eed7f2f0 bb088017}_{\mathrm{x}}$
$\texttt{163bc60d f45a0ecb 1bcd289b 06cbbfea 21ad08e1 847f3f73}_{\mathrm{x}}$
$\texttt{78d56ced 94640d6e f0d3d37b e67008e1 86d1bf27 5b9b241d}_{\mathrm{x}}$
$\texttt{eb64749a 47dfdfb9 6632c3eb 061b6472 bbf84c26 144e49c2}_{\mathrm{x}}$

Step 2. We use the following algorithm to enforce the first two security criteria.

1. for $i = 0$ to 63 do
   (a) while there exists $0 \leq j < i$ such that $\mathrm{RT}(j) = \mathrm{RT}(i)$, replace $\mathrm{RT}(i)$ by $\mathrm{RT}(i) + 1 \bmod 2^{\frac{m}{4}}$.
2. if KD is even, replace KD by KD + 1.
3. change the EES string accordingly so that Equation (14) holds.

Step 3. From this EES string we now define

$$\mathrm{EES} = \mathrm{KAB}_0|\ldots|\mathrm{KAB}_{15}|\mathrm{KS}. \tag{15}$$

Note that the third security criterion is necessarily satisfied, otherwise we would have collisions in RT.

At the end of the algorithm, we obtain a constant EES string depending on the parameters and which comes from the expansion of e and all the defined constants. We notice that for $m = 128$ all criteria are satisfied when EES is equal to the original expansion string of e (written in hexadecimal as above). For large m, it is highly unusual that we have to change it (but for KD with probability 1/2).

3 Benchmarks and Implementations

Straightforward implementations of DFC are quite slow on 32-bit microprocessors for the nominal choices of parameters, due to the critical operation $ax + b \bmod 2^{64} + 13$. Efficient implementations require non trivial tricks. That is why the original implementation of DFCv1, which was bound to NISTs requirements (namely, ANSI-C implementation, which restricts to 32-bit words and prohibits the use of the 32-bit *times* 32-bit $\rightarrow$ 64-bit multiplication of most processors), was quite slow and actually slower than most other candidates, especially since it dealed with endianess as well. The ANSI-C implementation required 3600 clock cycles per encryption (without key setup) on a Pentium Pro. This should be compared with the 392 clock cycles on the same processor using assembly language and processor specific tricks. Further implementation tricks (which were summarized by Noilhan [15]) and clever use of specific architectures of microprocessors have shown that DFC was among the fastest AES candidates, and notably the fastest one on ALPHA 64-bit microprocessors (310 clock cycles per encryption without the key setup, on an ALPHA 21164a in assembly code[4]).

DFCv2 does not introduce important implementation differences from DFCv1 for the nominal choice of the parameters. More precisely, only the key schedule has changed, and even the complexity of the key setup has not changed (it roughly takes four basic encryptions).

4 Security Analysis

4.1 Provable Security Results

We state the security results in terms of the new parameters (m, k, r, s).

Ideal key schedule. We recall that the security results consist, firstly of theoretical results for an ideal extension of DFCv2 in which the RK_i sequence is assumed to be uniformly distributed (we will call $\mathrm{DFCv2}^*(m, r)$ this ideal algorithm which does not depend on k or s), secondly of some practical results on the real DFCv2 algorithm in which we have to make a heuristic assumption stated below.

Theorem 1 ([24]). *The best advantage of an attack limited to two adaptively chosen plaintexts for distinguishing* $\mathrm{DFCv2}^*(m, r)$ *from a uniformly distributed random permutation is bounded by*

$$\mathop{\mathrm{BestAdv}}_{\mathrm{Cl}_a^2}(\mathrm{DFCv2}^*(m,r), C^*) \leq \frac{1}{2}\left(3\left(\left(\frac{p}{2^{\frac{m}{2}}}\right)^2 - 1\right) + \frac{8}{2^{\frac{m}{2}}}\right)^{\left\lfloor \frac{r}{3} \right\rfloor} \tag{16}$$

where p is the smallest prime number greater than $2^{\frac{m}{2}}$.

If we let $p = 2^{\frac{m}{2}}(1 + \delta)$, the previous upper bound can be approximated by

$$\frac{1}{2}\left(6\delta + 2^{3-\frac{m}{2}})\right)^{\left\lfloor \frac{r}{3} \right\rfloor}. \tag{17}$$

This shows that the best advantage is negligible against 2^{-m} if $r \geq 9$ when the attack is limited to two chosen plaintexts (i.e. in the $d = 2$ case). For $m = 128$, we have $\delta = 13.2^{-64}$ and we get back the bound of DFCv1

$$\mathop{\mathrm{BestAdv}}_{\mathrm{Cl}_a^2}(\mathrm{DFCv2}^*(128,r), C^*) \leq \frac{1}{2} 2^{-57.5\left\lfloor \frac{r}{3} \right\rfloor} \tag{18}$$

[4] Implementation due to Robert Harley, see [8]. See also [1,15].

From the decorrelation theory we know that the security against any attack limited to two chosen plaintexts implies the security against some reasonable formalization of differential and linear cryptanalysis (see [20]). Namely, the average complexity of differential cryptanalysis (over the distribution of the keys) needs at least to be within the order of 1/4BestAdv, as for the linear cryptanalysis (from an asymptotic bound). In this context, for instance, differential cryptanalysis can be formalized into:

1. pick a differential characteristic (a, b)
2. query an input pair of difference a until the corresponding output pair has a difference of b

It is well-known that this formalization is the core of regular differential cryptanalysis [3]. For instance, 2R attacks apply such a procedure on $r-2$ rounds. Since we can claim that the differential cryptanalysis core against $\mathrm{DFCv2}^*(128, 6)$ has a complexity of 2^{115}, we can thus claim that $\mathrm{DFCv2}^*(128, 8)$ is secure against a 2R differential cryptanalysis up to a complexity of 2^{115}.

Similarly, the average complexity of any known plaintext coming from an iterated attack of order one (*i.e.* an iterated attack in which each iteration extracts one bit of information from one known plaintext/ciphertext pair) needs to be at least within the order of $1/2\sqrt{\mathrm{BestAdv}}$ (see [22]).

More precisely, we recall the following result:

Theorem 2 ([20,22]). *For any differential distinguisher of complexity* n *against* $DFCv2^*(m, r)$*, the advantage* Adv_D *is such that*

$$\mathrm{Adv}_D \leq n\mathrm{BestAdv} + \frac{n}{2^m - 1} \tag{19}$$

where BestAdv is bounded by Equation (16). Similarly, for any linear distinguisher we have

$$\lim_{n \to +\infty} \frac{\mathrm{Adv}_L}{n^{\frac{1}{3}}} \leq 9.3 \left(4\mathrm{BestAdv} + \frac{1}{2^m - 1}\right)^{\frac{1}{3}}. \tag{20}$$

For any known plaintext iterated distinguisher of order 1 we have

$$\mathrm{Adv}_I \leq 3\left(\left(\frac{9}{2}2^{-m} + 3\mathrm{BestAdv}\right)n^2\right)^{\frac{1}{3}} + n\mathrm{BestAdv}. \tag{21}$$

Real key schedule. Since DFCv2 has a new key scheduling algorithm, we need to transform the security results on $\mathrm{DFCv2}^*$ to DFCv2. Let $\mathcal{D}(m, k, r, s)$ be the distribution of $(\mathrm{RK}_1, \ldots, \mathrm{RK}_r)$ spanned by the key scheduling algorithm of $\mathrm{DFCv2}(m, k, r, s)$ when K is a uniformly distributed k-bit key, and we let $\mathcal{D}^*$ denote the uniform distribution over rm-bit sequences. $\mathrm{DFCv2}^*$ relies on the $\mathcal{D}^*$ distribution, but DFCv2 uses the $\mathcal{D}$ distribution.

Let $H_t(m, k, r, s)$ be the best advantage of a Turing machine limited to t steps for distinguishing $\mathcal{D}(m, k, r, s)$ from $\mathcal{D}^*$ from a single sample (*i.e.* an rm-bit string). (H_t is a heuristic function. We need to assume that for a reasonable t, H_t is small.)

Theorem 3. *If for some class $Cl_{t,n}$ of distinguishers limited to a complexity of t and n oracle calls, the advantage for distinguishing* $\mathrm{DFCv2}^*(m,r)$ *from a random permutation is limited to BestAdv, then the advantage for distinguishing* $\mathrm{DFCv2}(m,k,r,s)$ *from a random permutation in class Cl is limited to* $H_{t+O(n)}(m,k,r,s) + \mathrm{BestAdv}$ *where the $O(n)$ corresponds to the cost of simulating DFCv2 on n oracle calls.*

Therefore, assuming that the complexity of a practical attack already includes an overestimated cost for simulating the oracle calls (in practice, using an oracle costs more than simulating it), then all security results on DFCv2* extend to DFCv2 with an advantage offset of H_t.

For practical t, $m \geq 128$, $k \geq 128$, $s \geq 4$ and $r \leq \frac{128}{s}$, we conjecture that $H_t(m,k,r,s)$ is negligible.

4.2 Best Attacks

So far, the best reported attack is Knudsen's impossible differential attack [9] against DFCv2 reduced to six rounds. It requires 2^{70} chosen plaintexts and a complexity of 2^{126} encryptions (see [10]). This attack can be compared to a 1R attack that uses a differential characteristic on 5 rounds (for which the complexity lower bound indicated by Theorem 2 is of order 2^{57} chosen plaintexts).

Harvey recently reported[5] an attack against four rounds which uses the non-injective properties of the round functions.

Another quite strong claim of insecurity is due to Rijmen and Knudsen [10]. Basically, they study a key-dependent one-round differential characteristic for a modified version of DFC and deduce some insecurity claims. One problem is that they use a difference which is not defined by the XOR operation but by the mod $2^{\frac{m}{2}}$ difference at the input and by the mod p difference at the output. This makes it hard to pile up such kinds of characteristics.

For instance, Rijmen and Knudsen noticed that if we replace all XORs in the round function by regular additions, every single input difference leads to about 800 possible output differences, one of it with probability 2^{-7} (with $m = 128$). These mod $2^{\frac{m}{2}}$ output differences translate into XOR output differences within a probability related to their Hamming weight (because of carry bits). We can thus estimate that the real DFC round function will lead to no key-dependent differential probabilities greater than 2^{-23}. Therefore, we believe the Rijmen-Knudsen observation does not imply any insecurity statement for DFCv2.

5 The DFC Controversy

The submission of DFCv1 to AES led to a controversy which was oriented towards three arguments which are addressed in the following subsections.

[5] at the Rump Session of Fast Software Encryption 2000.

5.1 Speed

DFCv1 was claimed to be among the slowest of the 15 AES candidates, and one of the worst for low-cost smart card implementations.

A fair performance comparison is a really hard task, as was shown by the AES conferences [18, section 4]. Timings have been collected by Granboulan [8] and Lipmaa [12], and DFC is without any doubt among the 8 fastest candidates in software : Crypton, DFC, E2, Mars, RC6, Rijndael, Serpent and Twofish. It is even the fastest candidate on architecture that have fast multiplication (Alpha and TurboSparc). When compared to the five finalists, DFC can be considered as achieving the same performances as Mars on current architectures (but being twice as fast on future architectures like Itanium). The dependence of DFC on multiplication can be compared to the dependence of RC6 on data dependent rotations.

In addition, it was shown in [17] that DFC was reasonably implementable on very simple embedded microprocessors (such as Motorola 6805 for smart cards). DFC does not take as much room on low-cost smart cards as Mars, and should have similar performances. On high-end smart cards (StrongARM) DFC is probably the fastest of all AES candidates.

In conclusion, DFC performances are not the best, but they compare very well to Mars, which is one of the finalists.

5.2 Provable Security

The provable security results were subject to controversy. We believe this was due to misunderstanding and we would like to clarify the situation.

After the DES was proposed, several other block ciphers showed up without any formal security argument. The security was essentially empirical: a block cipher was secure until someone came up with an attack. Although this approach proved very fruitful for promoting research on the analysis of block ciphers, the security provided is now debatable since the analysis time of all world experts is rather limited. Besides, we note that there were 15 candidates to analyze in less than one year, while DES weaknesses were discovered only after 10 years of public exposure.

Another tremendous amount of regular block ciphers use regular "security claims", which essentially consists of heuristic arguments (like the argument on H_t we used above for DFCv2). Typically, people argue that we cannot get good differential characteristics by regular active S-box counting arguments. This paradigm was inherited by the work of Biham and Shamir [3] and Coppersmith's analysis of DES [4].

In 1992, Lai and Massey [11] proposed the formal notion of "Markov cipher" which characterizes ciphers for which differentials can nicely be piled up. For these ciphers we can formally prove the heuristic security arguments against differential cryptanalysis on average over the key space.

Another more formal approach on which seldom block ciphers are based (including MISTY [13,14]) is inherited by Nyberg-Knudsen Theorem [16]. It consists of using ad hoc constructions with heavy non-linear constraints on S-boxes and deducing that the block cipher has no good differential property on average on the key distribution. These results are however limited to differential (and linear) attacks.

Our paradigm obtains similar results to the previous approach in a more general setting for basically no cost. It further provides more freedom in the construction of the block cipher. Thus, we believe it is a better alternative which follows the construction trends.

One objection by Rijmen and Knudsen [10] argued that since there exist insecure algorithms for which similar security claims hold, such claims are worthless. Indeed, the affine cipher $x \mapsto K_1x + K_2$ has a perfect pairwise decorrelation, which means that Theorem 2 holds with BestAdv= 0, and in particular, no differential distinguisher gets a relevant advantage. (The differential is chosen before the attack itself in this model, so it is independent on the key.) This comes from the fact that we can "only" say that the probability of any differential is low on average over the key space. Previous formal approaches suffer from the same drawbacks. Actually, the Markov cipher approach is quite similar, and the Nyberg-Knudsen approach has the same result. As compared to the Nyberg-Knudsen approach, the present one holds for regular ciphers (not only to ad hoc constructions). Therefore we claim that DFCv2 benefits from the all regular heuristic security arguments and the present formal security proof (which is not the case of the affine cipher, nor of any other regular cipher). This suggests that DFC has its raison d'être.

5.3 Security Margin

Another criticism against DFC was its low "security margin". The DFC philosophy consisted of not overestimating the minimal number of secure rounds and committing to the formal results obtained by decorrelation theory. We actually believe that for construction reasons, the security increases faster with the number of rounds than for other designs. We chose $r = 8$ as a challenge to the cryptographic community. Users who would not like to commit on such a bet can however freely use a higher number of rounds in the present DFCv2 version (for instance, $r = 12$ as recommended by Biham).

6 Conclusion

We have presented an updated version of DFC in which we changed the key schedule and introduced scalable parameters. These modifications left the security results unchanged (except the weak key attack which has been fixed).

Despite of the controversy during the AES process, we have shown that DFCv2 is one of the fastest block ciphers (on 64-bit microprocessors which have an optimized multiplier for $m = 128$) and benefits from some formal security results in addition to regular heuristic arguments.

Although this first generation of decorrelated ciphers may still be improved by the research community, we hope this paradigm will be useful to develop future cryptographic algorithms.

Acknowledgements

It was our pleasure to participate to the DFC team. For this we are deeply grateful to all other team members: Olivier Baudron, Henri Gilbert, Marc Girault, Helena Handschuh, Philippe Hoogvorst, Antoine Joux, David Pointcheval, Thomas Pornin, Guillaume Poupard and Jacques Stern, and particularly to Robert Harley who got interest in optimizing DFC implementations and who joined us.

We thank the CNRS, the École Normale Supérieure and France Telecom for sponsoring the project, particularly FIST (France Innovation Scientifique et Transfert), Jacques Stern and Henri Gilbert for devoting time to the initiative.

We wish to thank all the researchers who have got interest in DFC, particularly Don Coppersmith and Ron Rivest, and Dominik Behr, Brian Gladman, Danjel McGougan, Terje Mathisen, David Seal for their good optimizations.

We would finally like to thank NIST for having initiated this story and for having acknowledged us to participate.

References

1. O. Baudron, H. Gilbert, L. Granboulan, H. Handschuh, A. Joux, P. Nguyen, F. Noilhan, D. Pointcheval, T. Pornin, G. Poupard, J. Stern, S. Vaudenay. Report on the AES Candidates. In *Proceedings from the Second Advanced Encryption Standard Candidate Conference*, National Institute of Standards and Technology (NIST), Rome, Italy, March 1999.
2. O. Baudron, H. Gilbert, L. Granboulan, H. Handschuh, R. Harley, A. Joux, P. Nguyen, F. Noilhan, D. Pointcheval, T. Pornin, G. Poupard, J. Stern, S. Vaudenay. DFC Update. In *Proceedings from the Second Advanced Encryption Standard Candidate Conference*, National Institute of Standards and Technology (NIST), Rome, Italy, March 1999.
3. E. Biham, A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.
4. D. Coppersmith. The Data Encryption Standard (DES) and its Strength against Attacks. *IBM Journal of Research and Development*, vol. 38, pp. 243–250, 1994.
5. H. Feistel. Cryptography and computer privacy. *Scientific American*, vol. 228, pp. 15–23, 1973.
6. H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, S. Vaudenay. Decorrelated Fast Cipher: an AES Candidate. (Extended Abstract.) In *Proceedings from the First Advanced Encryption Standard Candidate Conference*, National Institute of Standards and Technology (NIST), Ventura, California, U.S.A., August 1998.
7. H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, S. Vaudenay. Decorrelated Fast Cipher: an AES Candidate. Submitted to the Advanced Encryption Standard process. In *CD-ROM "AES CD-1: Documentation"*, National Institute of Standards and Technology (NIST), August 1998.

8. L. Granboulan. AES ; Timings of the best known implementations. `http://www.di.ens.fr/~granboul/recherche/AES/timings.html`
9. L. R. Knudsen. DEAL - A 128-Bit Block Cipher. Submitted to the Advanced Encryption Standard process. In *CD-ROM "AES CD-1: Documentation"*, National Institute of Standards and Technology (NIST), August 1998.
10. L. R. Knudsen, V. Rijmen. On the Decorrelated Fast Cipher (DFC) and Its Theory. In *Fast Software Encryption*, Roma, Italy, Lectures Notes in Computer Science 1636, pp. 81–94, Springer-Verlag, 1999.
11. X. Lai. *On the Design and Security of Block Ciphers*, ETH Series in Information Processing, vol. 1, Hartung-Gorre Verlag Konstanz, 1992.
12. H. Lipmaa. AES Ciphers: speed `http://www.tcm.hut.fi/~helger/aes/`
13. M. Matsui. New Structure of Block Ciphers with Provable Security against Differential and Linear Cryptanalysis. In *Fast Software Encryption*, Cambridge, United Kingdom, Lectures Notes in Computer Science 1039, pp. 205–218, Springer-Verlag, 1996.
14. M. Matsui. New Block Encryption Algorithm MISTY. In *Fast Software Encryption*, Haifa, Israel, Lectures Notes in Computer Science 1267, pp. 54–68, Springer-Verlag, 1997.
15. F. Noilhan. Software Optimization of Decorrelation Modules. In *Selected Areas in Cryptography*, Kingston, Ontario, Canada, Lectures Notes in Computer Science 1758, pp. 175–183, Springer-Verlag, 2000.
16. K. Nyberg, L. R. Knudsen. Provable Security against a Differential Cryptanalysis. *Journal of Cryptology*, vol. 8, pp. 27–37, 1995.
17. G. Poupard, S. Vaudenay. Decorrelated Fast Cipher: an AES Candidate well suited for Low Cost Smart Cards Applications. In *CARDIS' 98*, Louvain-la-Neuve, Belgium, Lectures Notes in Computer Science 1820, pp. ???, Springer-Verlag, 2000.
18. B. Preneel *et al.* Comments by the NESSIE Project on the AES Finalists. Submitted to the Advanced Encryption Standard process, round 2 comments. National Institute of Standards and Technology (NIST), May 2000. `http://csrc.nist.gov/encryption/aes/round2/comments/20000524-bpreneel.pdf`
19. R.L. Rivest. The RC5 Encryption Algorithm. In *Fast Software Encryption*, Leuven, Belgium, Lectures Notes in Computer Science 1008, pp. 86–96, Springer-Verlag, 1995.
20. S. Vaudenay. Provable Security for Block Ciphers by Decorrelation. Invited talk. In *STACS 98*, Paris, France, Lectures Notes in Computer Science 1373, pp. 249–275, Springer-Verlag, 1998. Full Paper: technical report LIENS-98-8, Ecole Normale Supérieure, 1998. (`ftp://ftp.ens.fr/pub/reports/liens/`)
21. S. Vaudenay. Feistel Ciphers with L_2-Decorrelation. In *Selected Areas in Cryptography*, Kingston, Ontario, Canada, Lectures Notes in Computer Science 1556, pp. 1–14, Springer-Verlag, 1999.
22. S. Vaudenay. Resistance Against General Iterated Attacks. In *Advances in Cryptology EUROCRYPT'99*, Prague, Czech Republic, Lectures Notes in Computer Science 1592, pp. 255–271, Springer-Verlag, 1999.
23. S. Vaudenay. On the Lai-Massey Scheme. In *Advances in Cryptology ASIACRYPT'99*, Singapore, Lectures Notes in Computer Science 1716, pp. 8–19, Springer-Verlag, 2000.
24. S. Vaudenay. Adaptive-Attack Norm for Decorrelation and Super-Pseudorandomness. In *Selected Areas in Cryptography*, Kingston, Ontario, Canada, Lectures Notes in Computer Science 1758, pp. 49–61, Springer-Verlag, 2000.

25. S. Vaudenay. On Provable Security for Conventional Cryptography. Invited talk. (To appear in the proceedings of ICISC' 99, LNCS, Springer-Verlag.)

A Test Vector

A test vector for the nominal choice of parameters ($m = 128$, $k \in \{128, 192, 256\}$, $r = 8$ and $s = 4$) is included below.

We have chosen to use KS as key and 0_x as plaintext. We recall the value of KS:

$$\texttt{86d1bf27 5b9b241d eb64749a 47dfdfb9}_\text{x}$$
$$\texttt{6632c3eb 061b6472 bbf84c26 144e49c2}_\text{x}$$

The key schedule tests all KAB entries but KAB_1 and KAB_{12} (which are not used with this choice of parameters). It results in the following subkeys:

round	*subkeys*
1	05c5bd24 aa6ba7df 0846cb21 e1ab0dc7$_\text{x}$
2	63b67a97 142061ce c034fd75 ea2cd3d9$_\text{x}$
3	abf20d20 9b963b4c f04efdd6 2a6c459d$_\text{x}$
4	27215d71 2b28c6cb e2f472eb 288d47e8$_\text{x}$
5	02aae49f caf2ddf3 60405b1d d0d269a7$_\text{x}$
6	2a516cdc 6270af2b f3db8f26 c26ea9eb$_\text{x}$
7	94d3b898 ccbca828 4f6af189 39230738$_\text{x}$
8	6c9d3c7e d7059bcc 7a3d4288 f232b634$_\text{x}$

The iterated encryptions of plaintext 0_x tests all entries in the RT table for $j = 64$.

j	DFCv2^j_KS
0	00000000 00000000 00000000 00000000$_\text{x}$
1	1ba5af95 aba096ed 5b6c9750 2fe7efa2$_\text{x}$
2	0f36105c 1302d52a e47d6d42 dfaaf5c7$_\text{x}$
3	bb58f671 54c59d52 fefb03a8 74c138c5$_\text{x}$
4	acc4cf76 6505c09f 5ffe10d5 b021d66c$_\text{x}$
8	62395cc6 ba7bf158 f78b5897 04a1db59$_\text{x}$
16	387c4222 c61f5e69 7946e251 eb40031a$_\text{x}$
32	4ab38d66 16247c2a efbe6cde 4d302a86$_\text{x}$
64	ee043b7d a8610c46 3e282198 c93887b4$_\text{x}$