ON THE SECURITY OF THE $DEKART$ PRIMITIVE

Gilles Piret
UCL Crypto Group, Laboratoire de Microelectronique, Universite Catholique de Louvain, Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium.
piret@dice.ucl.ac.be

Francois-Xavier Standaert
UCL Crypto Group, Laboratoire de Microelectronique, Universite Catholique de Louvain, Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium.
fstandae@dice.ucl.ac.be

Gael Rouvroy
UCL Crypto Group, Laboratoire de Microelectronique, Universite Catholique de Louvain, Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium.
rouvroy@dice.ucl.ac.be

Jean-Jacques Quisquater
UCL Crypto Group, Laboratoire de Microelectronique, Universite Catholique de Louvain, Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium.
jjq@dice.ucl.ac.be

Abstract $DeKaRT$ primitives are key-dependent reversible circuits presented at CHES 2003. According to the author, the circuits described are suitable for data scrambling but also as building blocks for block ciphers. Data scrambling of internal links and memories on smart card chips is intended for protecting data against probing attacks. In this paper, we analyze the $DeKaRT$ primitive using linear cryptanalysis. We show that despite its key-dependent behavior, $DeKaRT$ still has strongly linear structures, that can be exploited even under the particular hypothesis that only one bit of the ciphertexts is available to the attacker (as it is the case in the context of probing attacks), and using very few plaintext-ciphertext pairs.
The attack methodology we describe could be applied to other data scrambling primitives exhibiting highly biased linear relations.

Keywords: Smart Card, Probing Attacks, Data Scrambling, Linear Cryptanalysis.

1. Introduction

Probing attacks on smart cards are invasive techniques consisting in introducing a conductor in some point of a tamper-resistant chip to monitor the electric signal, in order to recover secret information passing through this point [4, 5]. For example, the bus connecting the RAM and the microprocessor is particularly vulnerable. Using classical block ciphers like DES or AES seems to provide a natural solution to this problem; however it is simply not realistic, because of the very high throughput and small size requirements.

It is why primitives have been developed offering hastier and lighter solutions, but at the cost of a lower security level. In [2] implementations of keyed bit permutations are proposed, trying to simultaneously achieve small logical depth and large key space. Nevertheless this type of primitive has the drawback of being perfectly linear. At CHES2003, the $DeKaRT$ construction was presented [1], which was aimed at providing non-linearity but at the cost of a heavier structure than in [2].

In this paper, we analyze the security of the $DeKaRT$ primitive in accordance with the usual properties of block ciphers. Despite the complex structure of the primitive and its key-dependent behavior, it is underlined that the use of such a scrambling function does not efficiently prevent probing attacks.

In practice, we illustrate the strongly linear structure of the $DeKaRT$ block. The main observation is that, although the suggested block size of $DeKaRT$ primitives is very small compared to block ciphers (which reduces the total number of plaintext-ciphertext pairs a priori available), it is still possible to do linear predictions of its inputs. Moreover, these predictions may only involve a very limited number of output bits, what is actually relevant in the probing attack context. As a block cipher building block, $DeKaRT$ exhibit even stronger weaknesses as more linear relationships are available to the attacker.

2. Specification of a Concrete Instance of $DeKaRT$ and Notations

In [1] general building principles are given for construction of a $DeKaRT$ data scrambling function, but there is no precise instance defined. We believe it is a mistake, as security analysis requires a completely specified cipher; and security of a cipher which has not been subject to a substantial effort regarding security analysis can be questioned (except if a security proof is given, which is never the case for block ciphers). This is why we first specify a cipher, based on the design principles given in [1].

A *generic building block* acts on a small number of input data bits which are divided into two groups of m and n bits. The m input bits are used for control and are passed to the output intact, like in the Feistel structure. They are used to select k out of $2^m k$ key bits by the multiplexer (MUX) circuit with m control

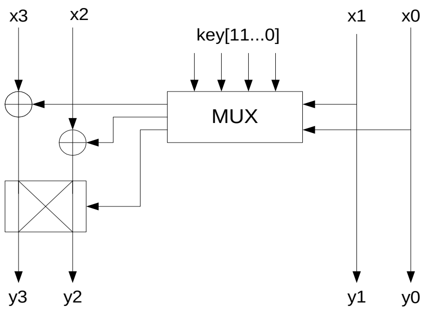

Figure 1. Elementary $DeKaRT$ building block.

bits, $2^m k$ input bits and k output bits. In the original paper, the author suggests an *elementary $DeKaRT$ building block* with parameters $(m, n, k) = (2, 2, 3)$ as shown in Figure 1 where after a XOR with 2 key bits, x_3 and x_2 pass through a conditional switch. We will use this building block in our specification. Each box requires 12 key bits. We will denote these bits by

$$\begin{aligned}(k^{(11)}, k^{(10)}, k^{(01)}, k^{(00)}) &= \\ &(k_X^{(11)}, k_{\oplus 1}^{(11)}, k_{\oplus 2}^{(11)};\, k_X^{(10)}, k_{\oplus 1}^{(10)}, k_{\oplus 2}^{(10)};\, k_X^{(01)}, k_{\oplus 1}^{(01)}, k_{\oplus 2}^{(01)};\, k_X^{(00)}, k_{\oplus 1}^{(00)}, k_{\oplus 2}^{(00)})\end{aligned}$$

where $k^{(i)}$ denotes the three key bits selected by control bits $(x_1, x_0) = i$; $k_X^{(i)}$ is conditioning the switch, while $k_{\oplus 1}^{(i)}$ is XORed with x_3 and $k_{\oplus 2}^{(i)}$ is XORed with x_2. The set of such 12 key bits is called a *subkey*.

The instance of $DeKaRT$ we will consider acts on blocks of 16 bits. Thus the key dependent layer, denoted KT for ***K**eyed **T**ransform*, consists in the parallel application of 4 such elementary blocks. The number of rounds considered is 5 (this number is given as an example in [1], p.104). We will denote the subkeys parameterizing the four elementary blocks of the r^{th} round by $RK_3^r, RK_2^r, RK_1^r, RK_0^r$, from left to right. The set of 4 such subkeys is called a *round key* (thus it has 48 bits).

The KT layer alternates with a ***B**it **P**ermutation* layer (noted BP). Two design rules regarding BP were given in [1]:

- The control bits ((x_1, x_0) in Fig. 1) in each layer should be used as the transformed bits ((x_3, x_2) in Fig. 1) in the next layer.
- For each elementary block, input bits should come from the maximum possible number of blocks in the previous KT layer. In the instance considered, it means that each of the 4 input bits to an elementary block comes from a different block in the previous KT layer (which also implies that the 4 output bits of an elementary block are sent to 4 different

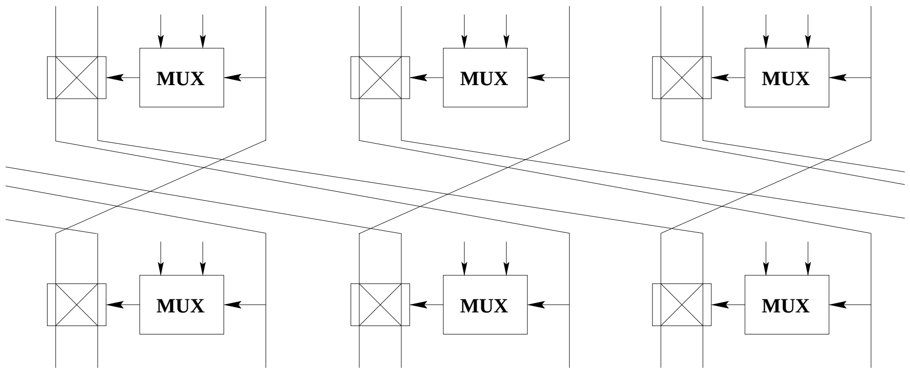

Figure 2. a part of the Key Expansion

blocks in the next KT layer).

We arbitrarily chose a bit permutation following these rules. It is given in Annex A.1 (see also Fig. 3).

Finally, as suggested in [1], the key expansion algorithm also alternates “keyed" layers (with the key being an arbitrarily chosen constant), with bit permutations layer. The 48-bit data obtained every two rounds is used as a round key. The keyed layer is made out of the parallel application of 16 *reduced DeKaRT building block* (as proposed in [1]) with 3-bit input and output, alternating with a bit permutation layer complying with the design rules given above. Fig. 2 pictures part of this algorithm. Note that the bit permutation layer could have been chosen less “regular", but in fact its influence on the results of our attack is negligible.

3. Analysis of an elementary $\boldsymbol{DeKaRT}$ block

A $DeKaRT$ building block generates key-dependent boolean functions. If a (2,2,3) block is used, 4096 substitution tables (4×4 bits) can be generated, as this type of block is parameterized by 12 key bits.

In this section, we investigate the possible linear approximations of an elementary $DeKaRT$ block. For each output bit, there exist $2^4 - 1$ possible non-trivial input masks (i.e. $2^4 - 1$ possible linear combinations of input bits) and if we combine output bits together, we have $(2^4 - 1) \times (2^4 - 1)$ possible non-trivial linear approximations of the $DeKaRT$ block. For such a small block size, the problem of finding good linear approximations is therefore easily solved by exhaustive search.

Table 1. Linear approximations of a single $DeKaRT$ block.

ϵ	1/2	3/8	1/4
Nbr. Approximations	2304	1024	768

In this paper, we define the *bias* of a linear approximation that holds with probability p as $\epsilon = |p - 1/2|$. We also denote a linear approximation with probability $p = 0$ or $p = 1$ as a *perfect* linear approximation.

As the $DeKaRT$ block defines 4096 substitution tables, we first investigated the best linear approximations of each individual table. It is summarized in Table 1, where we rejected approximations involving bits y_0, y_1 only as they are obviously perfectly linear. We observe that more than one half of the generated tables presents perfect linear approximations and may therefore be considered as very weak from a cryptographic point of view.

On the basis of this first experiment, we may therefore assess that a large number of keys will generate weak building blocks for the scrambling function or block cipher as they are perfectly approximated by a linear approximation. This motivated a more general analysis.

4. The Attack

The scenario of the attack is the following: we consider the case where the data scrambling function $DeKaRT$ is used to protect communication between the smart card microprocessor and the RAM; this is the most common use for data scrambling functions. We assume the attacker is allowed to play with the microprocessor, which implies that he can send known data to the memory. Moreover he has access to a small number of bits of the encrypted data via probing attack. Formally, this means that the attacker can obtain a certain number m of pairs (P, c), where P is a known plaintext, and c is one fixed bit of the corresponding ciphertext C. His goal is to obtain information about a secret data (such as an RSA private key) present in the card and read from the RAM at some time. Thus it it **not** a key recovery attack, in the sense that we do not intend to retrieve the key used for $DeKaRT$ data scrambling (it can be changed at each use of the card anyway). This security model is the one described in [2].

Due to the building blocks of $DeKaRT$ being implemented using key-dependent MUXs, the probability of a linear relation between a given bit of the ciphertext and some bits of the plaintext is highly key-dependent. In our attack, pairs (P, c) are used to identify a linear relation between one only bit of the ciphertext and a linear combination of bits of the plaintext, that holds with a high bias for the key used. Then when the secret data pass through the channel between the RAM and the processor, the relation we just identified permits probabilistic information about it to be retrieved.

Let λ be the linear relation we are considering. Knowing m pairs (P, c), we count how many of these satisfy the linear relation λ; let n_λ^m be the random variable corresponding to this number. We would like to compute the probability that λ holds for a random plaintext, provided n_λ^m takes value B:

$$P_{K,P}[\lambda \text{ holds} | n_\lambda^m = B] \tag{1}$$

$|P_{K,P}[\lambda \text{ holds}|n_\lambda^m = B] - 1/2|$ gives the reliability of the prediction we will make. Let us define the random variable N_λ as being the number of plaintexts for which λ holds, out of all 2^{16} plaintexts this time. Then we have:

$$\begin{aligned} P_{K,P} &\ [\lambda \text{ holds}|n_\lambda^m = B] \\ &= \sum_{A=0}^{2^{16}} P[\lambda \text{ holds}|N_\lambda = A] \cdot P[N_\lambda = A|n_\lambda^m = B] \\ &= \sum_{A=0}^{2^{16}} A/2^{16} \cdot P[N_\lambda = A|n_\lambda^m = B] \\ &= \sum_{A=0}^{2^{16}} A/2^{16} \cdot P[n_\lambda^m = B|N_\lambda = A] \cdot \frac{P[N_\lambda = A]}{\sum_{A'=0}^{2^{16}} P[N_\lambda = A'] \cdot P[n_\lambda^m = B|N_\lambda = A']} \\ &\cong \frac{\sum_{A=0}^{2^{16}} A/2^{16} \cdot P[Bi(m, A/2^{16}) = B] \cdot P[N_\lambda = A]}{\sum_{A'=0}^{2^{16}} P[Bi(m, A'/2^{16}) = B] \cdot P[N_\lambda = A']} \end{aligned} \tag{2}$$

where $Bi(.,.)$ denotes the binomial distribution law, and the last approximation assumes $m \ll 2^{16}$. Thus computation of (1) requires knowledge of the probability distribution of N_λ, i.e. $\{P_K[N_\lambda = A]\}_{A=0}^{2^{16}}$. Next section will show how such a distribution can be computed.

5. Computing Probability Distribution for a Linear Relation Through a 5-Round Cipher

Consider 5 rounds of $DeKaRT$, beginning and ending with a keyed layer (see Fig. 3). We denote the plaintext by $(p_{15}, p_{14}, ..., p_1, p_0)$ or $(a_{15}^{(1)}, a_{14}^{(1)}, ..., a_1^{(1)}, a_0^{(1)})$. Also:

$$KT((a_{15}^{(i)}, a_{14}^{(i)}, a_{13}^{(i)}, ..., a_1^{(i)}, a_0^{(i)})) =: (b_{15}^{(i)}, b_{14}^{(i)}, b_{13}^{(i)}, ..., b_1^{(i)}, b_0^{(i)})$$

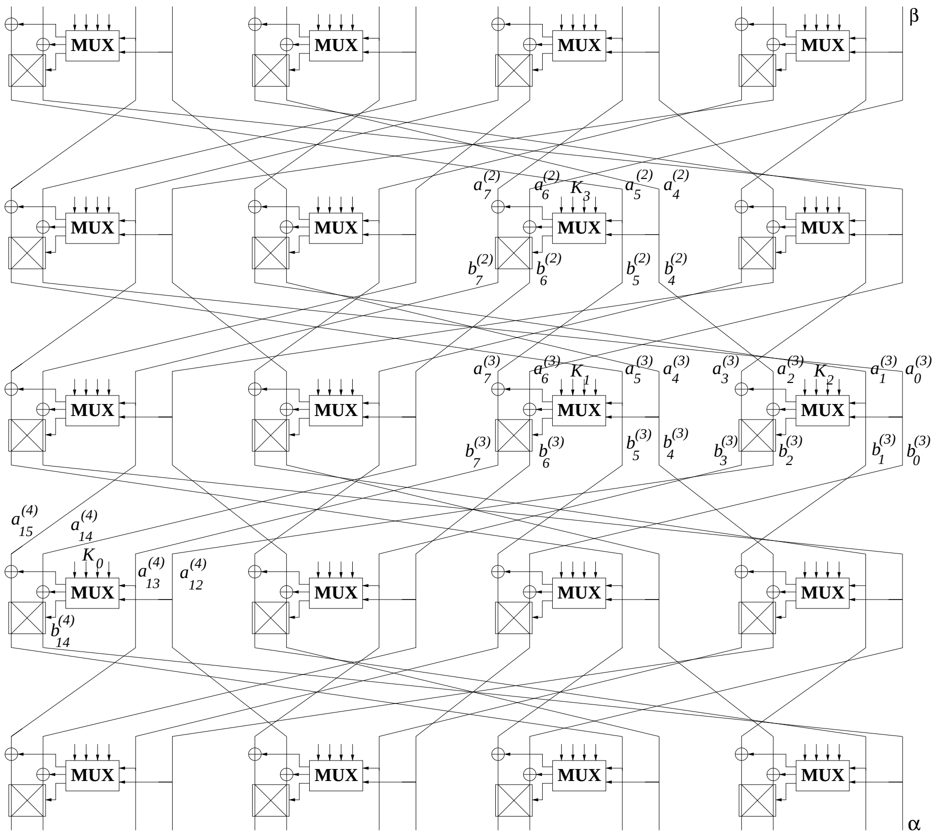

Figure 3. Linear Approximation through a 5-Round $DeKaRT$

And:

$$BP((b_{15}^{(i)}, b_{14}^{(i)}, b_{13}^{(i)}, ..., b_1^{(i)}, b_0^{(i)})) =: (a_{15}^{(i+1)}, a_{14}^{(i+1)}, a_{13}^{(i+1)}, ..., a_1^{(i+1)}, a_0^{(i+1)})$$

The exponent denotes the round number and the ciphertext will be denoted by

$$(b_{15}^{(5)}, b_{14}^{(5)}, ..., b_1^{(5)}, b_0^{(5)}) \text{ or } (c_{15}, c_{14}, ..., c_1, c_0)$$

Consider one bit $\alpha := b_0^{(5)}$ of the ciphertext after a 5-round cipher. As an example, we will analyze the linear relation between α and $\beta := a_0^{(1)}$ (see Fig. 3). Other relations can be analyzed similarly. We write successively:

- $\alpha = b_{14}^{(4)}$
- $b_{14}^{(4)}$ is a function of $(a_{15}^{(4)}, a_{14}^{(4)}, a_{13}^{(4)}, a_{12}^{(4)})$ depending on $K_0 := RK_3^4$ (see Fig. 3).

- $(a_{15}^{(4)}, a_{14}^{(4)}) = (b_7^{(2)}, b_6^{(2)})$
- $a_{13}^{(4)} = b_7^{(3)}$ is a function of $(a_7^{(3)}, a_6^{(3)}, a_5^{(3)}, a_4^{(3)})$ depending on $K_1 := RK_1^3$. As for β fixed $a_6^{(3)}$ is balanced (i.e. takes values 0 and 1 equally often) and as this bit affects α only by means of the block keyed by K_1, key bits $k_{\oplus 2}^{(i)}$ of K_1 do not affect the probability of equation $\alpha = \beta$.
- $a_{12}^{(4)} = b_2^{(3)}$ is a function of $(a_3^{(3)}, a_2^{(3)}, a_1^{(3)}, a_0^{(3)})$ depending on $K_2 := RK_0^3$. For the same kind of reason as before, key bits $k_{\oplus 1}^{(i)}$ of K_2 do not affect the probability.
- Finally, $(b_7^{(2)}, b_6^{(2)})$ is a function of $(a_7^{(2)}, a_6^{(2)}, a_5^{(2)}, a_4^{(2)})$ depending on $K_3 := RK_1^2$. Still using the same arguments, we note that key bits $k_{\oplus 1}^{(i)}$ of K_3 do not affect the probability.

Thus the probability of $\alpha = \beta$ (computed over all 2^{16} plaintexts) depends on the 4 subkeys K_0, K_1, K_2, K_3 (or at least part of them). We write them as:

$$K_0 = (k_0^{(11)}, k_0^{(10)}, k_0^{(01)}, k_0^{(00)})$$
$$K_1 = (k_1^{(11)}, k_1^{(10)}, k_1^{(01)}, k_1^{(00)})$$
$$K_2 = (k_2^{(11)}, k_2^{(10)}, k_2^{(01)}, k_2^{(00)})$$
$$K_3 = (k_3^{(11)}, k_3^{(10)}, k_3^{(01)}, k_3^{(00)})$$

There are $12 + 3 \times 8 = 36$ subkey bits implied. A priori this does not allow easy exhaustive computation of the probability for every key (complexity $2^{36} \times 2^{16}$). However there are groups of values for which the associated probability is the same. Let us say that 2 subkeys K_1 and K_{1*} are *equivalent* if for any K_0, K_2, K_3, the probability associated to (K_0, K_1, K_2, K_3) and (K_0, K_{1*}, K_2, K_3) is identical; Equivalence between 2 subkeys K_2 and K_{2*} is defined similarly. We can make the following observations:

- For β fixed $(a_5^{(3)}, a_4^{(3)})$ is balanced. Therefore two subkeys K_1 and K_{1*} such that there exists a permutation $\pi : \{0,1\}^2 \rightarrow \{0,1\}^2$ satisfying $k_1^{(i)} = k_{1*}^{(\pi_i)} (\forall i \in \{00, 01, 10, 11\})$ are equivalent.
- The same argument can be used for K_2.
- As from the output of the block keyed by K_1 only bit $b_7^{(3)}$ matters, if $k_{1,X}^{(i)} = 1$ (for some $i \in \{00, 01, 10, 11\}$), then $k_{1,\oplus 1}^{(i)}$ does not affect the probability. Otherwise stated, if K_1 and K_{1*} are such that

$k_1^{(i)} = (1,0,0)$ while $k_{1*}^{(i)} = (1,1,0)$ (other bits being the same), they are equivalent.

- With the same kind of argument, $k_2^{(i)} = (1,0,0)$ and $k_{2*}^{(i)} = (1,0,1)$ are equivalent.
- Similarly, as output $b_{15}^{(4)}$ of the block keyed by K_0 does not matter, we have:

$$k_0^{(i)} = (0,0,0) \sim k_0^{(i)} = (0,1,0) \qquad k_0^{(i)} = (0,0,1) \sim k_0^{(i)} = (0,1,1)$$
$$k_0^{(i)} = (1,0,0) \sim k_0^{(i)} = (1,0,1) \qquad k_0^{(i)} = (1,1,0) \sim k_0^{(i)} = (1,1,1)$$

- Suppose $\exists i,j : k_1^{(i)} = (0,0,0)$ and $k_1^{(j)} = (0,1,0)$. Then, the key bit added to $a_7^{(3)}$ is 0 for the 2^{14} plaintexts for which $(a_5^{(3)}, a_4^{(3)}) = i$; it is 1 for the 2^{14} plaintexts for which $(a_5^{(3)}, a_4^{(3)}) = j$. Thus (taking into account that for $(a_5^{(3)}, a_4^{(3)})$ fixed $a_7^{(3)}$ is balanced) when $(a_5^{(3)}, a_4^{(3)}) \in \{i,j\}$, $b_7^{(3)} = 1$ one half of the times.
Now if we replace $k_1^{(i)}$ and $k_1^{(j)}$ by $(1,0,0)$, when $(a_5^{(3)}, a_4^{(3)}) \in \{i,j\}$ we have $b_7^{(3)} = a_6^{(3)}$. As $a_6^{(3)}$ is balanced (for fixed $(a_5^{(3)}, a_4^{(3)})$), we still have that $b_7^{(3)} = 1$ one half of the times.
The conclusion is that if a given key is such that $\exists i,j : k_1^{(i)} = (0,0,0)$ and $k_1^{(j)} = (0,1,0)$, then if $k_1^{(i)}$ and $k_1^{(j)}$ are replaced by $(1,0,0)$ the key obtained is equivalent to the former one.

- Similarly, if $\exists i,j : k_2^{(i)} = (0,0,0)$ and $k_1^{(j)} = (0,0,1)$, replacing these bits by $k_2^{(i)} = k_2^{(j)} = (1,0,0)$ does not change the probability.

Putting all these observations together, there are 9 equivalence classes for K_1 as well as for K_2. They are given in Table 2, with the number of elements in each class (out of 2^{12}).

Finally, the number of different quadruples (K_0, K_1, K_2, K_3) to explore in order to compute the probability distribution $\{P_K[N_{\alpha=\beta} = A]\}_{A=0}^{2^{16}}$ is $9^2 \cdot (2^8)^2 \cong 2^{22}$. This number could be further reduced, by exploiting more complex equivalences such as: "if K_0 has such value, then value of K_1 does not matter". However it is not necessary as with complexity $2^{22} \cdot 2^{16}$, the probability distribution of $N_{\alpha=\beta}$ is computable. It is roughly given in Annex A.2.

It is worth mentioning that in the previous discussion we made the (classical) hypothesis that the round keys are independent and uniformly distributed, while in practise they are derived from the master key using the key expansion

Table 2. Equivalence classes for K_1 and K_2 with their cardinalities

K_1	#	K_2	#
$(000; 000; 000; 000)$	16	$(000; 000; 000; 000)$	16
$(000; 000; 000; 010)$	448	$(000; 000; 000; 001)$	448
$(000; 000; 000; 100)$	128	$(000; 000; 000; 100)$	128
$(000; 000; 010; 010)$	1120	$(000; 000; 001; 001)$	1120
$(000; 000; 010; 100)$	896	$(000; 000; 001; 100)$	896
$(000; 010; 010; 010)$	448	$(000; 001; 001; 001)$	448
$(000; 010; 010; 100)$	896	$(000; 001; 001; 100)$	896
$(010; 010; 010; 010)$	16	$(001; 001; 001; 001)$	16
$(010; 010; 010; 100)$	128	$(001; 001; 001; 100)$	128

Table 3. Families of linear relations with the best mean bias

Mean bias	Number of lin. rel.	Output bit
$7 \cdot 10^{-2}$	16	$\in S_1$
$4 \cdot 10^{-2}$	64	$\in S_2$
$2,5 \cdot 10^{-2}$	192	$\in S_1$
$1,5 \cdot 10^{-2}$	64	$\in S_2$

described in section 2; in fact some quadruples (K_0, K_1, K_2, K_3) simply cannot be derived from a master key. Computation of the value taken by $N_{\alpha=\beta}$ for a small number of random keys from which round keys are derived perfectly validated the hypothesis.

6. Searching for other Linear Relations Through a 5-Round Cipher

The procedure described in the previous section to compute the distribution of N_λ for a given linear relation λ is complicated and has non-negligible time complexity. It is however possible to identify linear relations having a big mean bias by evaluating this bias using only a part of the 2^{16} plaintexts, and this for a relatively small number of keys. Doing this, we observed that there are “families" of linear relations having about the same mean bias. Moreover linear relations from some families have their output bit belonging to $S_1 \equiv \{c_0, c_1, c_4, c_5, c_8, c_9, c_{12}, c_{13}\}$, while those from the other families have their output bit belonging to $S_2 \equiv \{c_2, c_3, c_6, c_7, c_{10}, c_{11}, c_{14}, c_{15}\}$. This is due to the fact that the last round of $DeKaRT$ need not be approximated if the output bit $\in S_1$.

Details about the families with the best mean bias are given in Table 3.

The linear relations of the first family (with bias $\sim 7 \cdot 10^{-2}$) are given in Table 4.

Table 4. Linear relations through 5-round $DeKaRT$ with the highest mean bias

$p_0 \oplus c_0$	$p_4 \oplus c_1$	$p_8 \oplus c_8$	$p_{12} \oplus c_9$
$p_0 \oplus c_5$	$p_4 \oplus c_4$	$p_8 \oplus c_{13}$	$p_{12} \oplus c_{12}$
$p_1 \oplus c_1$	$p_5 \oplus c_0$	$p_9 \oplus c_9$	$p_{13} \oplus c_8$
$p_1 \oplus c_4$	$p_5 \oplus c_5$	$p_9 \oplus c_{12}$	$p_{13} \oplus c_{13}$

7. Implementation of the Attack

As explained in Section 4, we assume that the attacker knows for example 128 pairs $\{(P^j, c_i^j)\}_{j=0\ldots127}$, where P^j is a plaintext and c_i^j is the i^{th} bit of the corresponding ciphertext. One attack strategy could be:

1 Consider all $2^{16} - 1$ possible input masks μ (i.e. all possible linear combinations of input bits).

2 For each of them, compute the bias of $\mu \bullet P = c_i$ over the 128 pairs ($\bullet$ denotes the scalar product over $\mathbb{Z}_2^{16}$).

3 The attacker intercepts a ciphertext bit c_i^* of which he does not know the corresponding plaintext P^*. The input mask μ^* with the highest bias (computed at step 2) is used to predict the unknown bit $\mu^* \bullet P^*$.

The efficiency of this algorithm is measured by the bias associated to μ^*, **computed over all 2^{16} plaintexts** this time. Indeed, it gives the reliability of the guess made at step 3. Practical experiments show that we have a mean bias of 0,059 when the ciphertext bit considered $\in S_1$ and of 0,022 when it is in S_2.

However it is possible to do better if in step 1 of the attack, we restrain ourself to the 336 relations mentioned in Section 6 (or more precisely, to those of them concerning bit c_i). Then the bias obtained is 0,107 when the ciphertext bit considered $\in S_1$ and 0,074 when it is in S_2. Moreover the probability computed over 128 plaintexts almost always "goes in the same direction" than the one computed on all 2^{16} plaintexts (i.e. suggests the same value for $\mu \bullet P \oplus c_i$). This significant improvement is due to the fact that if we consider all possible input masks, it is often the case that estimation on 128 plaintexts happens to emphasize a linear relation which in fact has a small (or null) bias when computed over all 2^{16} plaintexts; a pre-selection of “a priori good" input masks greatly reduces this phenomenon. It is this improvement that motivated the research of *a priori* good linear relations described in Section 6.

In Table 5 we give mean biases for different numbers of pairs (P^j, c_i^j) known by the attacker. We insist on the fact that these figures are *mean* biases. This means that sometimes the bias associated to μ^* will be 0, which

Table 5. Mean bias as a function of the number of pairs known by the attacker

# pairs	if $c_i \in S_1$	if $c_i \in S_2$
64	0,095	0,067
128	0,107	0,074
256	0,118	0,083
512	0,123	0,087
1024	0,127	0,092

means that the attack has completely failed. Other times the bias will be 1/4 (or even 1/2) and the information gained by the attacker is real. The attacker must be able to compute a priori the bias he can expect. It is given by equation (2) in Section 4.

As an improvement to the attack, it could also be possible to consider several approximations (implying the same ciphertext bit c_i) simultaneously in order to retrieve more information. However this is far from trivial, as the possible correlation between these approximations must be taken into account. The paper from Biryukov*&al.* [6] could help in this context.

Also, we assumed only one bit of the ciphertext was available. If several are, this allows more bits of information about the plaintext to be retrieved (moreover linear approximations implying linear combinations of these ciphertext bits can be considered, which can improve the efficiency of the attack).

8. Conclusion

In this paper we have seen that $DeKaRT$, despite its structure being significatively more complex than previous primitives, is vulnerable to linear cryptanalysis. Even using one only bit of the ciphertext (as it is often the case in the context of probing attacks), it is possible to obtain information about an unknown plaintext using very few known (Plaintext, Ciphertext bit) pairs. We do not claim $DeKaRT$ is useless for data scrambling: indeed, the requirements for such a type of primitive can be relaxed in comparison with usual requirements for block ciphers. More than the overall structure, some proposals for the number of rounds provided in the original paper seem to be too optimistic for a really strong security. The purpose of this paper is rather showing that such type of key-dependent transform still has strongly linear structures. It is why we believe it is possible to construct a better primitive with the same throughput and size constraints; probably a structure nearer the classical paradigms of block cipher design (constant and highly non-linear S-boxes) could achieve it. There is place for research effort in this direction.

Table A.1. Probability Distribution of $N_{\alpha=\beta}$

$N_{\alpha=\beta}$	P_K
$\in [0, 16383]$	0,006
= 16384	0,010
$\in [16385, 22527]$	0,029
= 22528	0,013
$\in [22529, 24575]$	0,013
= 24576	0,053
$\in [24577, 26623]$	0,024
= 26624	0,028
$\in [26625, 27647]$	0,009
= 27648	0,022
$\in [27649, 28671]$	0,014
= 28672	0,045
$\in [28673, 29695]$	0,012
= 29696	0,026
$\in [29697, 30719]$	0,013
= 30720	0,050
$\in [30721, 31743]$	0,015
= 31744	0,016
$\in [31745, 32767]$	0,013
= 32768	0,178

Appendix

1. The Bit Permutation layer BP

The bit permutation we chose is:

Inp. Bit Pos.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Outp. Bit Pos.	5	0	15	10	1	4	11	14	13	8	7	2	9	12	3	6

2. Probability Distribution of $N_{\alpha=\beta}$

Out of the $2^{16} + 1$ a priori possible values for $N_{\alpha=\beta}$, only 199 occur with a non-zero probability. In Table A.1 we only mention the ones having probability $\geq 0,01$. Moreover we give probabilities associated with intervals. As it is easy to show that $P[N_{\alpha=\beta} = A] = P[N_{\alpha=\beta} = 2^{16} - A]$, the table only goes from 0 to 2^{15}.

References

[1] J.D.Golic, *DeKaRT: A New Paradigm for Key-Dependent Reversible Circuits*, Proceedings of CHES 2003, Lecture Notes in Computer Science, vol. 2779, pp. $98 - 112$, 2003.

[2] E. Brier, H. Handschuh, C. Tymen, *Fast Primitives for Internal Data Scrambling in Tamper Resistant Hardware*, Proceedings of CHES 2001, Lecture Notes in Computer Science, vol.

2162, pp. $16 - 27$, 2001.

[3] M. Matsui, *Linear Cryptanalysis Method for DES Cipher*, Advances in Cryptology - EUROCRYPT 93, Lecture Notes in Computer Science, vol. 765, pp. $386 - 397$, 1994.

[4] R. Anderson and M. Kuhn, *Tamper resistance - a Cautionary Note*, second USENIX Workshop on Electronic Commerce Proceedings, pp. $1 - 11$, Oakland, California, November 1996.

[5] O. Kөmmerling and M. Kuhn, *Design principles for Tamper-Resistant Smartcard Processors*, USENIX Workshop on Smartcard Technology, Chicago, Illinois, USA, May 1999.

[6] A. Biryukov, C. De Canniere, M. Quisquater, *On Multiple Linear Approximations*, Available at http://eprint.iacr.org/, 2004/057.