Cryptanalysis of White Box DES Implementations*

Louis Goubin, Jean-Michel Masereel, and Michaël Quisquater

Versailles St-Quentin-en-Yvelines University
45 avenue des Etats-Unis
F-78035 Versailles Cedex
{Louis.Goubin,Jean-Michel.Masereel,Michael.Quisquater}@uvsq.fr

Abstract. Obfuscation is a method consisting in hiding information of some parts of a computer program. According to the Kerckhoffs principle, a cryptographical algorithm should be kept public while the whole security should rely on the secrecy of the key. In some contexts, source codes are publicly available, while the key should be kept secret; this is the challenge of code obfuscation. This paper deals with the cryptanalysis of such methods of obfuscation applied to the DES. Such methods, called the "naked-DES" and "nonstandard-DES", were proposed by Chow *et al.* [5] in 2002. Some methods for the cryptanalysis of the "naked-DES" were proposed by Chow *et al.* [5], Jacob *et al.* [6], and Link and Neuman [7]. In their paper, Link and Neuman [7] proposed another method for the obfuscation of the DES.
In this paper, we propose a general method that applies to all schemes. Moreover, we provide a theoretical analysis. We implemented our method with a C code and applied it successfully to thousands of obfuscated implementations of DES (both "naked" and "non-standard" DES). In each case, we recovered enough information to be able to invert the function.
Keywords: Obfuscation, cryptanalysis, DES, symmetric cryptography, block cipher.

1 Introduction

In recent years, the possibility of obfuscating programs has been investigated. From a theoretical point of view, Barak *et al.* [1] have proven impossibility results for the task of obfuscating computer programs. In particular, it turns out that there exists a family of programs such that: on the one hand each program is non learnable (i.e. its execution does not give any information about its original source code), but on the other hand every obfuscator (i.e. the program producing an obfuscation) fails completely when given any program of this family as input. However it has not been proved that specific instances, particularly cryptographic primitives, are impossible to obfuscate.

* This work has been supported in part by the French ANR (Agence Nationale de la Recherche), through the CrySCoE project, and by the région Île-de-France

In 2002, Chow *et al.* [4,5] suggested two different obfuscations, one for the AES, the other for the DES. The AES obfuscation was cryptanalysed by Billet *et al.* [2,3] in 2004. Chow *et al.* [5] also mounted an attack on their first DES obfuscation version (called "naked-DES"). Jacob *et al.* [6] and Link and Neuman [7], proposed two other attacks on the "naked-DES". Here, breaking the "naked-DES" means recovering the secret key.

A second version of DES obfuscation, called "nonstandard-DES", was given by Chow *et al.* [5]. This "nonstandard-DES" is obtained by obfuscating the usual DES composed with initial and final secret permutations. In this context, breaking such a "nonstandard-DES" implementation means recovering the secret key and the secret initial and final permutations.

Moreover, many industrial actors have developed obfuscated implementations of cryptographic algorithms, in particular for DRM, Pay-TV, and intellectual property protection. (e.g. cloakware [12], retroguard [13], Yguard [14]).

This paper is structured as follows : In Section 2, we give an overview of the obfuscation methods given by Chow *et al.* and by Link and Neumann. Section 3 is devoted to our attack on the "naked-DES". In Section 4, we adapt our attack to the "non standard" DES. Section 5 is devoted to our implementation of this attack. In Section 6, we compare our attack to the one of Wyseur *et al.* [11]. Finally, we conclude in Section 7. All proofs are available in the appendices.

2 DES Obfuscation Methods

Chow *et al.* [5] proposed two types of DES obfuscation. The first one, called "naked-DES", produces an usual DES. The second one, called the "nonstandard-DES", is a slight modification of the standard DES algorithm. This last version is the one they recommend.

Let us describe the "naked-DES" (see Figure 7). The standard DES is implemented by means of many functions. The first one is an affine function M_1, which is the composition of the initial permutation, the expansion (slightly modified in order to duplicate all the 32 right bits), and a bit-permutation $\phi_0 : \mathbb{F}_2^{96} \to \mathbb{F}_2^{96}$. The role of ϕ_0 is to send 48 bits to the corresponding S-box entries, the 48 remaining bits being sent randomly to the T-box entries (see Figure 8). Eight of these T-boxes are derived from the eight S-boxes of the DES (see Figure 1), and the four remaining T-boxes are identities (or more generally bit permutations, see Figure 8 (T_{12})). An affine function $M_{2,1}$ follows the T-boxes. This affine function is the composition of the P and Xor operation of the standard DES, and a bit-permutation ϕ_1 (see Figure 7). Each of the 16 rounds is the composition of the T-boxes and an affine function $M_{2,i}$. The last round is followed by an affine function M_3 which is the composition of a selection function, and the final permutation. This function takes for arguments the outputs of the affine function $M_{2,16}$ of the last round and returns the ciphertext (see Figure 7).

We will denote by A_i, one of these components (T_i, M_1, $M_{2,i}$ or M_3). The obfuscator program computes numbers of random nonlinear permutations on

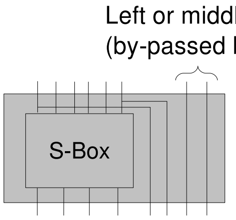

Fig. 1. T-Box

$\mathbb{F}_2^s$, $b_{k,l}$ ($s = 4$ or 8). These permutations are referred by Chow *et al.* [5] as io-block encoding bijections. Twenty-four or twelve of these io-block encoding bijections are concatenated in order to obtain nonlinear permutations on $\mathbb{F}_2^{96}$, $P_{i,j}$. Each component A_i is obfuscated between permutations $P_{i,1}$ and $P_{i,2}$. The resulting functions $P_{i,1} \circ A_i \circ P_{i,2}$ are stored in arrays in order to be used by the obfuscated program. When considering consecutive components, the final permutation of the first component, and the initial permutation of the second component, cancelled out (see Figure 7) i.e. :

$$(P_{i,1} \circ A_i \circ P_{i,2}) \circ (P_{j,1} \circ A_j \circ P_{j,2}) = P_{i,1} \circ (A_i \circ A_j) \circ P_{j,2} \ .$$

This "naked-DES" was cryptanalysed by the authors themselves [5].

In order to repair the scheme, they proposed the "nonstandard-DES". It consists in adding two affine bijections M_0 and M_4 before and after the "naked-DES", respectively (see Figure 7). It is not specified by Chow *et al.* [5] whether M_0 and M_4 are block encoded (i.e. respectively preceded and followed by nonlinear random permutations). In this paper, we consider that M_0 and M_4 are not block encoded.

Further improvement on the attack of the "naked-DES" were given by Link and Neumann [7]. They suggested another solution which consists in merging the T-boxes and the affine function $M_{2,i}$ of each round. This way, we do not have access to the T-boxes outputs. Moreover, the $M_{2,i}$ functions of the different rounds are block encoded in another way.

In this paper, we describe an attack that defeats both "nonstandard-DES" and the Link and Neumann's schemes.

3 Attack on the "Naked-DES"

As mentioned before, the "naked-DES" proposed by Chow *et al.* [5] was already cryptanalysed in the papers [5,6,7]. In this section, we show how to cryptanalyse the improved version of the "naked-DES" proposed by Link and Neumann [7]. Note that our method also works for the "naked-DES" proposed by Chow *et al.* [5]. In what follows, we will denote by "regular DES", the one described in the standard [10] (without PC1), and we will use the same notations.

Our attack is divided into two phases and is based on a truncated differential attack. Roughly speaking, the first phase consists in generating pairs of messages (X,X') such that the right part of the images, through IP and the first round of a regular DES, are equal (for a given key K) (see Figure 2.b). The second phase consists in evaluating those pairs of messages (X, X') on the "naked-DES" and in checking a condition that we specify below. The pairs that satisfy the test provide a key candidate.

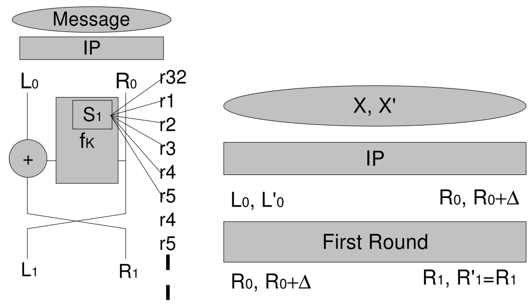

Fig. 2. One round of DES, and attack principle

Let us go into the details. Remember that $f(., K)$ denotes the function of the regular DES, we will also denote it by $f_K(.)$ (see Figure 2.a). Let X be an initial message, (L_0, R_0) denotes its image through IP, and (L_1, R_1) is the image of (L_0, R_0) through the first round, i.e. $(L_1, R_1) = (R_0, L_0 \oplus f(R_0, K))$. Consider a function f, vectors X and Δ, the derivative $f(X) \oplus f(X \oplus \Delta)$ will be denoted by $D_\Delta f(X)$. Let us first motivate our algorithm. Let K be a fixed unknown key. Assume we want to find the first round 6-bit subkey corresponding to S_i (for the sake of clarity, we will restrain ourselves to $i = 1$). We generate candidate keys such that only the 6 key bits of S_1 of the first round are modified. For each of these keys, we compute pairs of messages (X,X') such that,

1. $\Delta = R_0 \oplus R'_0$ is zero, except for the second and third bits.
2. $L'_0 = L_0 \oplus D_\Delta f_K(R_0)$

Observe that the second and third bits of R_0 only affect the output of S_1 (see Figure 2.a) . Therefore, $f(R_0, K)$ and $f(R'_0, K)$ are identical except for the four bits corresponding to the output of S_1.

Under these conditions, in the next round we have $R_1 = R'_1$ and $L'_1 (= R'_0)$ is identical to $L_1 (= R_0)$ except for at most two bits. Consider now these two messages X and X' applied to the "naked-DES" with the correct key candidate. We observe that these bits (non-zero bits of $L'_1 \oplus L_1$) influence at most two io-block encoding bijections $b_{i,3}$ and $b_{j,3}$ (see Figure 8). If the key candidate is wrong, we will have $R_1 \neq R'_1$. Therefore many bits will change at the output of $M_{2,1}$, and we will be able to distinguish this situation from the correct key guess.

Here is an overview of the attack:

- Randomly choose a message X.
- Compute $(L_0, R_0) = IP(X)$ with IP public.
- Choose Δ such that only the second and third bits are different from 0.
- For any possible candidate value of 6-bit subkey:
  - Compute $L'_0 = L_0 \oplus D_\Delta f_K(R_0)$.
  - Compute $X' = IP^{-1}(L'_0, R_0 \oplus \Delta)$.
  - Apply X and X' to the obfuscated DES and save the values Y and Y' at the end of the first round.
  - Compare Y and Y' and compute in how many io-block encoding bijections they differ.
  - Reject the candidate if this number is strictly greater than 2. Otherwise, the candidate is probably correct.

This way, we can recover the 48 key bits of the first round of the DES. The 8 remaining bits are found by exhaustive search.

Remark 1. This algorithm can produce more than one candidate for the 6-bit subkey. It will provide wrong 6-bit subkeys in two situations.

1. Due to the balance property of the S-boxes and the fact they map six bits to four bits, four different inputs produce the same output. Therefore for each S-box, three wrong 6-bit subkeys will produce the same output as the correct key. To avoid this problem, we can launch this algorithm with another random initial message, or simply another Δ. In fact, we only have to change the values of the bits of R_0 and Δ corresponding to the input of S_1 (the bits 32,1,...,5). Actually, we can choose different pairs (X, X') such that the intersection of the key candidates associated to each of them is the correct key.
2. The second one is due to a propagation phenomena. Suppose we have a wrong 6-bit subkey producing a wrong S_1 output. It means that there are more than three bits of difference between (L_1, R_1) and (L'_1, R'_1). These differences could be mapped to the same io-block encoding bijection, leading to the flipping of only two io-block encoding bijections at the output of $M_{2,1}$. In this case, we launch this algorithm with several values for R_0. It leads to several lists of key candidates and the correct key belongs to the intersection. This way, wrong keys will be discarded.

4 Attack on the "Nonstandard-DES"

This section is dedicated to an attack on the "nonstandard-DES". Remind that the "nonstandard-DES" is a "naked-DES" where the affine functions M_1 and M_0 are replaced by $M_1 \circ M_0$ and $M_4 \circ M_3$ respectively (where M_0 and M_4 are mixing bijections, see Chow *et al.* [5]). As mentioned before, we assume that the inputs of $M_1 \circ M_0$ (respectively the outputs of $M_4 \circ M_3$) are not io-block encoded. Note that all the other functions are io-block encoded using bijections on $\mathbb{F}_2^4$ (the same principle applies for the obfuscation proposed by Link and Neuman [7] where the bijections are defined on $\mathbb{F}_2^8$). Moreover, we assume that the T-Boxes follow the same ordering in the different rounds. In what follows, we will not consider IP (inside M_1) w.l.o.g, for the sake of clarity.

In what follows, the term *preimage* will implicitly refer to the preimage with respect to the linear bijection M_0. Moreover, we say that a bit of a vector is *touching* an io-block encoding bijection if this bijection depends on this bit. Similarly, we will say that a vector *touches* an S-Box if non-zero bits touch it.

Our attack on the "nonstandard-DES" is based on the one on the "naked-DES". Our approach is based on a truncated differential attack. It consists in computing the images of a random vector X_0 at different levels in the obfuscated DES. We compare these values (called *initial-entries*) to the corresponding images of $X_0 \oplus \Delta$, where Δ satisfies some conditions depending on the context. This approach allows providing information about the key and the matrix M_0^{-1}, gradually. The full key and the matrix M_0^{-1} will be known at the end of the process. The way we store information about M_0^{-1} consists in considering lists of candidates for preimages of unspecified canonical vectors. Lists of candidates containing only one vector are called *distinguished* lists. This vector is then a column of M_0^{-1}. Note that these lists are actually vector spaces and can be shared by several canonical vectors. In practice, a list E will be shared by $\dim E$ canonical vectors (that are not necessary specified). Our algorithm works sequentially and consists in specifying these canonical vectors and shortening the lists. Our method can therefore be understood as a "filtering process". The different filters are described below.

Section 4.1 describes a preliminary step almost independent of the structure of the block cipher. It consists in finding vector spaces associated to a particular io-block encoding bijection at the input of the first round. This step allows getting global information about M_0^{-1}.

Section 4.2 describes a set of filters intending to refine information about M_0^{-1}. These steps are highly related to the studied block cipher. The first filter, described in Section 4.2, allows distinguishing lists that are associated to canonical vectors belonging either to right bits or left bits of the input of the first round (L_0 or R_0). The second filter, described in Section 4.2, extracts all the lists (marked as "right" in the previous filter) touching a single S-box (we will see that these lists play an important role). The third filter, described in Section 4.2, gathers the lists (marked as "left" in the previous filter) in sets associated to the output of S-boxes. Section 4.2 links T-Boxes (obfuscation of the keyed S-boxes) to S-Boxes. This information allows the last filter, presented in Section 4.2, to precisely specify the 1-to-1 link between the lists (marked as "left") and the (left) canonical vectors.

Section 4.3 explains how to extract the key and how to recover the full invertible matrices M_0^{-1} and M_4.

4.1 Block Level Analysis of $M_1 \circ M_0$

Recovering of the B_k's. Denote by K_k the space $(\{0\}^{4k-4} \times \mathbb{F}_2^4 \times \{0\}^{96-4k})$, and by $\overline{K}_k$, the space $(\mathbb{F}_2^{4k-4} \times \{0\}^4 \times \mathbb{F}_2^{96-4k})$. In what follows, the vector space spanned by a set of vectors S will be denoted $\langle S \rangle$. Also, e_i denotes the ith canonical vector (the position of the "one" is computed from the left and start from one) of the vector space $\mathbb{F}_2^{64}$. The sets $\{e_i \in \mathbb{F}_2^{64} \mid i = 1 \dots 32\}$ and $\{e_i \in \mathbb{F}_2^{64} \mid i = 33 \dots 64\}$ will be denoted by $\mathcal{S}_L$ and $\mathcal{S}_R$, respectively.

Ideally, we are looking for 24 vector spaces such that their vectors influence only one io-block encoding bijection at the output of $M_1 \circ M_0$. This would allow modifying only the input of one particular io-block encoding bijection. Unfortunately, due to the duplication of the bits in M_1 (because of the expansion E) this goal is impossible to reach. We will therefore try to approximate this situation and deal with the drawbacks afterwards. First we will have to give some notations, definitions and properties.

Denote by $F : \mathbb{F}_2^{64} \to \mathbb{F}_2^{96}$ the obfuscation of $M_1 \circ M_0$ (see Figure 7).

Let X be a vector in $\mathbb{F}_2^{96}$. Denote by π_k the projection $\pi_k : (\mathbb{F}_2^4)^{24} \to \mathbb{F}_2^4 : X = (x_1, \dots, x_{24}) \mapsto x_k$. Let b_k be the k^{th} io-block encoding bijection at the output of $M_1 \circ M_0$. The function F is written as

$$F(X) = (b_1 \circ \pi_1 \circ M_1 \circ M_0(X), b_2 \circ \pi_2 \circ M_1 \circ M_0(X), \dots, b_{24} \circ \pi_{24} \circ M_1 \circ M_0(X)) \ .$$

Definition 1. *Let k be an integer, $k \in [1, 24]$. We denote by $\mathcal{B}_k$ the vector space $\{X \in \mathbb{F}_2^{64} \mid \pi_k \circ M_1(X) = 0\}$. In other words, it is the subspace of vector X such that for any non-zero component e_i of X, $M_1(e_i)$ does not touch b_k, i.e. $\mathcal{B}_k = \langle e_j \mid \pi_k \circ M_1(e_j) = 0 \rangle$.*

Definition 2. *Let k be an integer, $k \in [1, 24]$. We denote by $\mathcal{E}_k$ the subspace of vector X such that for any non-zero component e_i of X, $M_1(e_i)$ touches b_k, i.e. $\mathcal{E}_k = \langle e_j \mid \pi_k \circ M_1(e_j) \neq 0 \rangle$.*

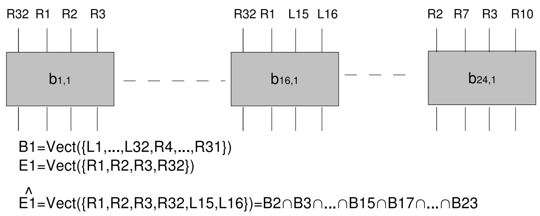

Fig. 3. Example

Remark 2. Note that $\mathbb{F}_2^{64}$ is the direct sum of $\mathcal{B}_k$ and $\mathcal{E}_k$ for any k, i.e. $\mathbb{F}_2^{64} = \mathcal{B}_k \bigoplus \mathcal{E}_k$.

We will denote by B_k the subspace $M_0^{-1}(\mathcal{B}_k)$, and by E_k the subspace $M_0^{-1}(\mathcal{E}_k)$.

Proposition 1. *For any k integer, $k \in [1, 24]$, $B_k = \{\Delta \in \mathbb{F}_2^{64} \mid D_\Delta F(\mathbb{F}_2^{64}) \subset \overline{K}_k\}$, the probability that Δ belongs to B_k, when Δ is randomly chosen, is greater or equal to $\frac{1}{2^4} = \frac{1}{16}$, and $60 \leq \dim(B_k) < 64$.*

Combining Definition 2 and Property 1, the vector space E_k can be described as the set of vectors Δ such that for any vector $X_0 \in \mathbb{F}_2^{64}$, $M_0(X_0) \oplus M_0(X_0 \oplus \Delta)$ has in total at most four non-zero components e_i, all of them touching the k^{th} io-block encoding bijection through M_1. Due to Property 1, it is easier to recover a basis for B_k's, than for E_k's. That is why we will first recover all the B_k's. Using Property 1, we only have to compute $D_\Delta F(X_0)$ for random $\Delta \in \mathbb{F}_2^{64}$ and determine to which space $\overline{K}_k$ it belongs. Using B_k's, we will recover E_k's, or at least, 24 vector spaces $\widehat{E}_k$ containing E_k with minimal dimension.

Recovering of the $\widehat{E}_k$'s. Let us now explain how to recover $\widehat{E}_k$. First, let us remark that for any $X \in \mathbb{F}_2^{64}$ and for any $\Delta \in \mathbb{F}_2^{64}$, we have $D_\Delta F(X) \in \overline{K}_k$ if and only if $D_\Delta \pi_k \circ M_1 \circ M_0(X) \in \overline{K}_k$. Let us introduce the following lemma.

Lemma 1. *Let k be an integer belonging to $[1, 24]$. If $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$ for any integer j distinct from k belonging to $[1, 24]$, then*

$$\mathcal{E}_k = \bigcap_{j \neq k} \mathcal{B}_j \ .$$

Since M_0 is a bijection, this lemma means that if $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$ for any integer $j \in [1, 24]$ different from k, then $E_k = \bigcap_{j \neq k} B_j$. Nevertheless, due to the bit-duplication, there exist indices k and j such that $\mathcal{E}_j \cap \mathcal{E}_k \neq \{0\}$ (and then $E_j \cap E_k \neq \{0\}$). Denote by J_k the set $\{j \mid \mathcal{E}_j \cap \mathcal{E}_k = \{0\}\}$, by $\widehat{\mathcal{E}}_k$ the subspace $\bigcap_{j \in J_k} \mathcal{B}_j$, and by $\widehat{E}_k$ the subspace $\bigcap_{j \in J_k} B_j$ where k is an integer belonging to $[1, 24]$.

Proposition 2. *For any integer $k \in [1, 24]$, $\mathcal{E}_k \subseteq \widehat{\mathcal{E}}_k$.*

Let us introduce a property that will allow us to give another characterization of J_k.

Proposition 3. *For any integer $i \in [1, 24]$ and for any integer $j \in [1, 24]$*

$$\dim(\mathcal{E}_i \cap \mathcal{E}_j) = 64 + \dim(B_i \cap B_j) - \dim(B_j) - \dim(B_i) \ .$$

A straightforward application of this property to the definition of J_k leads to $J_k = \{j \in [1, 24] \mid 64 = \dim(B_j) + \dim(B_k) - \dim(B_k \cap B_j)\}$. This characterization will be useful in order to compute $\widehat{E}_k$. If $\dim(\widehat{E}_k) + \dim(B_k) > 64$ then $E_k \subsetneq \widehat{E}_k$, and we have found a vector space containing strictly the one we search. Note that when $\dim(\widehat{E}_k) + \dim(B_k) = 64$, $E_k = \widehat{E}_k$. This case is particularly interesting because it reduces the complexity of the full cryptanalysis.

4.2 Bit Level Analysis of M_0^{-1}

In the previous section, we were looking for differences Δ associated to a specific io-block encoding bijection. It allowed us to get some information about M_0^{-1}. In this section, we refine our search and this will allow us to get enough information about M_0^{-1} in order to apply our method on the "naked-DES" to "nonstandard-DES". Our algorithm works sequentially and consists in a "filtering process". The different filters are described below.

Search for Candidates for Preimages of Elements Belonging to the Sets $\mathcal{S}_L$ and $\mathcal{S}_R$. Consider Δ be an element of $\mathbb{F}_2^{64}$ such that $M_0(\Delta) = e_i$ and $e_i \in \mathcal{S}_L$. The only non-zero bit of $M_1 \circ M_0(\Delta)$ touches only one io-block encoding bijection (recall that we do not consider IP). Therefore, Δ belongs to a single $\widehat{E}_k$. Assume now that $\Delta \in \mathbb{F}_2^{64}$ such that $M_0(\Delta) = e_i$ and $e_i \in \mathcal{S}_R$ then $M_1 \circ M_0(\Delta)$ has exactly two non-zero bits that may touch the same or two distinct io-block encoding bijections or equivalently Δ belongs to one or two spaces $\widehat{E}_k$. In what follows, we will call *double* an element $\Delta \in \mathbb{F}_2^{64}$ such that $M_0(\Delta) \in \mathcal{S}_R$ and the two non-zero bits of $M_1 \circ M_0(\Delta)$ touch the same io-block encoding bijection. For example, on Figure 8, the bit R_2 could be a double, since its two instances are in the input of T_1. By considering intersections between the spaces $\widehat{E}_k$, taken pairwise, we can distinguish preimages of elements of $\mathcal{S}_R$ from doubles or preimages of elements of $\mathcal{S}_L$.

Note that the intersections between spaces $\widehat{E}_k$ taken pairwise provide more information. Indeed, $\widehat{E}_i \cap \widehat{E}_j$ contains preimages of unknown canonical vectors. In particular, if $\dim(\widehat{E}_i \cap \widehat{E}_j) = 1$ then $\widehat{E}_i \cap \widehat{E}_j = \langle M_0^{-1}(e_k) \rangle$ for some k. In this case, we already know the preimage of an unknown canonical vector. When $\dim(\widehat{E}_i \cap \widehat{E}_j) > 1$ we can still take advantage of this fact even if it requires some extra searches.

Recovering Middle Bits. In order to apply our attack presented in Section 3, we need to exactly know the preimage of canonical vectors touching only a single S-Box of the first round (e.g. Right bits 2,3,6,7,10, ...). In what follows, we will refer to such a canonical vector as a *middle bit.* If a middle bit is not a double, then its two copies touch two different io-block encoding bijections. The first copy is in input of an S-box, leading to at least two bits of difference at the end of the first round of a regular DES, and 4 bits in our case, due to the expansion. The second copy is a by-passed bit (see Figure 1), leading to only one bit of difference at the end of the first round. Consider the bold path in Figure 8 starting from R_3 bit, in order to have a global view. Let us explain how we use this property.

Recall that X_0 is the initial-vector defined in Section 4. For each difference Δ belonging to the lists marked as input of the studied T-box, we apply $X_0 \oplus \Delta$ to the obfuscated DES by making an *injection fault.* This means that we set the input of this T-box to the initial-entry while we keep the input of the other T-Boxes (see Figure 4). We evaluate the number of io-block encoding bijections at the output of the first round that differs from the corresponding initial-entries. If only one io-block encoding bijection (at the output of the first round) differs from the corresponding initial-entry, we deduce that Δ could be the preimage of a middle bit. Therefore, a list containing preimages of several canonical vectors can be divided into two shorter lists; one list containing preimages of middle bits while the other contains preimages of non-middle bits.

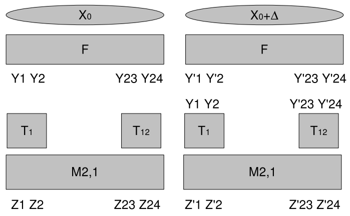

Fig. 4. Injection fault

Remark 3. If a T-box is touched by more than three middle bits or left bits, we deduce that this T-box does not contain any S-box. Note also that doubles can only be preimages of middle bits. Finally, a T-box touched by a double contains necessarily an S-box.

Recovering Left Bits. In order to apply our attack presented in Section 3, we need to know which group of four canonical vectors are xored with the output of each S-box of the first round. First, we determine the io-block encoding bijections that are touched by the outputs of the studied S-box and we denote by BS this set of bijections. In Figure 8, we can see that $BS = \{b_{1,3}, b_{3,3}, b_{8,3}, b_{12,3}, b_{15,3}, b_{20,3}, b_{24,3}\}$ for the S-box S_1. The elements $b_{i,3}$ of BS are characterised by $D_{\Delta_m} b_{i,3} \circ \pi_i \circ M_{2,1} \circ T \circ M_1 \circ M_0(X_0) \neq 0$, for all Δ_m belonging to a list marked as a middle bit of the studied S-box. Then, we store in an extra list $\mathcal{L}$ each Δ marked as left bits touching exactly two bijections of BS. This list contains all the preimages associated to canonical vectors that are potentially xored with the output of the S-box. Finally, we find $\Delta_l \in \langle \mathcal{L} \rangle$ such that for any bijection $b_{i,3} \in BS$ we have $D_{\Delta_m \oplus \Delta_l} b_{i,3} \circ \pi_i \circ M_{2,1} \circ T \circ M_1 \circ M_0(X_0) = 0$, where Δ_m belongs to a list marked as a middle bit of the studied S-box. This process is repeated with different Δ_m or X_0, until we find four linearly independent Δ_l or equivalently the vector space spanned by the preimages of the searched canonical vectors. We then compute the intersection between this space and all the lists. It allows us to split some of them in shorter lists (the intersection and the complementary space of the intersection). It may lead to lists containing a single vector (distinguished list).

Chaining. In this section, we will try to determine precisely the correspondence between T-boxes and S-boxes. Due to the remark in Section 4.2, we know which are the T-boxes containing an S-box. The probability that a selected T-box, denoted by T_1, contains S_1 is $1/8$. We determine the two T-Boxes that are touched by a canonical vector associated to a list marked as "right bit", "non-middle bit" and associated to T_1. Selecting one of these T-Boxes randomly, the probability that it contains S_2 is $1/2$. Out of the set of unselected T-Boxes, we select the one that is touched by a canonical vector associated to a list marked as "right bit", "non-middle bit" and associated to the previous selected T-Box. We continue the process until all T-Boxes have been selected (see Figure 5). Note that the probability to determine the right correspondence is $1/8 \times 1/2 = 1/16$.

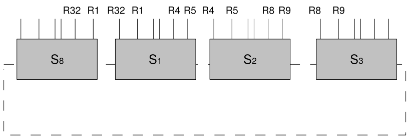

Fig. 5. Chaining

Bit Positions. At this stage, we have recovered between others, 32 preimages corresponding to unspecified left canonical vectors. In order to determine the correspondence, we use the following observation on the DES:

Out of the four left bits that are xored with the output of a specified S-Box, exactly two become (in the second round) middle bits.

Now, we just have to apply each of the preimages to the obfuscated DES and check whether the image of this vector in front of the second round is a middle bit (cf. 4.2). Assuming that the T-Boxes follow the same ordering in the different rounds, preimages corresponding to a middle bit (resp. non-middle bit) can be distinguished by observing the indices of the touched T-Boxes.
For example, for the first S-box, among the preimages of the four identified left canonical vectors,

- one of such vectors is the preimage of e_{23} (resp. e_{31}) if it is the preimage of a middle bit of S_6 (resp. S_8) in the second round.
- one of such vectors is the preimage of e_9 (resp. e_{17}) if it is not the preimage of a middle bit and it is in the input of S_2 and S_3 (resp. S_4 and S_5) of the second round.

4.3 The Attack

In Section 4.2, we have shown how to recover all the preimages of the left canonical vectors. In other words, we have recovered half of M_0^{-1} (columns and their positions). Also, some of the lists marked as middle bits contain only one vector but their corresponding canonical vector is however unknown. Therefore, some columns of M_0^{-1} are known up to their positions. Finally, the remaining lists marked as middle bits contain preimages of some canonical vectors $e_{i_1}, \ldots, e_{i_n}$ (their number is the dimension of the vector space spanned by the list). In this case, we select linearly independent vectors in the list and we associate each of them to one of the canonical vector e_{i_j}. Therefore, we are in the context of the attack of the "naked-DES" up to some adaptations. In particular, we have to choose X_0 belonging to the vector space spanned by the known columns of M_0^{-1}. The evaluation of the first round on $X_0 \oplus \Delta$ may lead to some difficulties. Indeed, we have to choose Δ belonging to the preimage of middle bits space which is not necessarily included in the vector space spanned by the known columns of M_0^{-1}. It turns out that we have to try all the candidates for this part of the matrix M_0^{-1}. For each of these candidates, we mount an attack like we did on the "naked-DES", which provides 48-bit key candidates. Note that wrong keys may be recovered. More importantly, here may be no key for this candidate for this part of the matrix M_0^{-1}. In other words, it means that we have to discard this candidate.

In order to determine the remaining part of M_0^{-1} (columns associated to non-middle bits), we apply a similar principle that we used for the "naked-DES". Indeed, we know the key and we know that for the "naked-DES" for any initial-message X_0 there always exists a difference Δ with non-zero right component such that the right part of the differential (evaluated in X_0) of the first round is zero. It means that in the context of the "nonstandard-DES", wrong candidates for M_0^{-1} can be discarded. Denote by K the space spanned by the known columns of the candidate for M_0^{-1} and by U the unknown columns of the candidate for M_0^{-1}. We have $K \oplus U = \mathbb{F}_2^{64}$. The candidate for M_0^{-1} can be discarded if there exists $X_0 \in K$ such that there does not exist Δ with a non zero-component in U such that the right part of the differential (evaluated in X_0) is zero.

At this stage, we have a 48-bit key candidate and a candidate for M_0^{-1}. We make an exhaustive search in order to determine the 8 remaining bits of the key. For each of them we try to solve a linear system in order to find the matrix M_4. If there is no solution for M_4 we deduce that the 8-bit key candidate is wrong. If all the 8-bit key candidates are wrong, we discard this particular M_0^{-1}. Note that this method also works if M_4 has io-block encoding bijections at its output.

Attack on Link and Neumann obfuscation: Our methods only use the outputs of the first and second round. In particular, we never use the outputs of the T-boxes. Therefore, our two attacks ("naked-DES", and "nonstandard-DES") can be applied on the Link and Neumann [7] obfuscation method. The only difference is that we will deal with larger lists.

5 Results

This attack was implemented with a C code. At each stage of the attack, the number of candidates decreases both for the key and for M_0^{-1}. Finally, it will lead to a unique 48-bit key candidate, a unique M_0^{-1} candidate, and a unique M_4 candidate. We have tested our attack on thousands of randomly generated obfuscated implementations of DES (both "naked" and "nonstandard" DES). Figure 6 shows the necessary time to complete the attack. We can observe that 95% of the attacks require less than 50 seconds, and 75% less than 17 seconds. The mean time is about 17 seconds. However, the attacks were executed on a standard PC. The code was not optimized and the performance can be further improved.

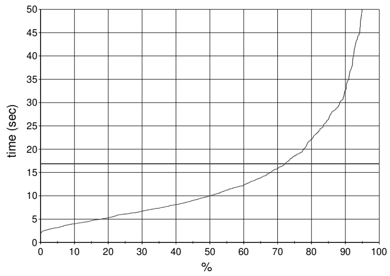

Fig. 6. Repartition of the attacks durations

6 Comparison to Wyseur *et al.*'s Work

In this section, we try to clarify the differences between our paper and the one of Wyseur *et al.* [11]. The main advantage of their method is that they are able to recover the key for the "nonstandard-DES" even when the transformations M_0 and M_4 are nonlinear. Nevertheless, they do not recover these transformations, and the only knowledge of the key is useless. They claim that the problem is straightforward in the linear case provided the key is known. As far as we know and without any additional tricks, this issue requires the use of algorithms that allow finding the linear bijections A and B satisfying the equation $G = A \circ F \circ B$, given the nonlinear functions F and G. These algorithms are exponential in the number of variables of F (see Patarin *et al.* [9]). Our method recovers these transformations in a short amount of time, when they are linear. The nonlinear case is still an open problem. Finally, Wyseur *et al.* [11] consider an obfuscation where the ϕ_i's have a restricted shape. While our model is unrestricted, they consider only ϕ_i's where all middle-bits touch only the four trivial T-boxes. It is not obvious whether their methodology can be adapted to the general case.

7 Conclusion

In this paper, we have given new techniques of cryptanalysis for the current obfuscation methods of DES. These techniques rely on a theoretical analysis and have also been implemented as a C program. We have implemented our method with a C code and have applied it successfully to more than a thousand obfuscated implementations of DES (both "naked" and "nonstandard" DES). All the studied instances have lead to a unique candidate for the DES key and for the M_0 and M_4 secret linear transformations. The key and the two linear transforms have been obtained within 17 seconds in average.

References

1. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S.P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology - CRYPTO '01, Aug. 19-23, 2001*, pages 1–18. Springer-Verlag, 2001
2. O. Billet. *Cryptologie Multivariable* Ph.D. thesis University of Versailles, December 2005.
3. O. Billet, H. Gilbert, and C. Ech-Chatbi. Cryptanalysis of a white box AES implementation. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 227–240. Springer, 2004.
4. S. Chow, P. Eisen, H. Johnson, and P. van Oorschot. White-box cryptography and an AES implementation. In 9^{th} *Annual Workshop on Selected Areas in Cryptography, Aug.15-16 2002*, volume 2595 of *LNCS*, pages 250–270. Springer-Verlag, 2002.
5. S. Chow, H. Johnson, P. van Oorschot, and P. Eisen. A white-box DES implementation for DRM applications. In *Proceedings of ACM CCS-9 Workshop DRM, Nov.18 2002*, volume 2696 of *LNCS*, pages 1–15. Springer-Verlag, 2003.
6. M. Jacob, D. Boneh, and E. Felten. Attacking an obfuscated cipher by injecting faults. In *Proceedings 2002 ACM Workshop on Digital Rights Management, November 18, 2002, Washington DC, USA.*, 2002.
7. H.E. Link and W.D. Neumann. Clarifying obfuscation: Improving the security of white-box encoding. 2004. http://eprint.iacr.org/.
8. J. Patarin and L. Goubin. Asymmetric cryptography with S-boxes. In *Proc. 1st International Information and Communications Security Conference*, pages 369–380, 1997.
9. J. Patarin, L. Goubin and N. Courtois Improved Algorithms for Isomorphisms of Polynomials. In *Proceedings of Eurocrypt'98*, volume 1403 of *LNCS*, pages 184–200. Springer-Verlag, 1998.
10. http://www.itl.nist.gov/fipspubs/fip46-2.htm
11. B. Wyseur, W. Michiels, P. Gorissen, and B. Preneel. Cryptanalysis of White-Box DES Implementations with Arbitrary External Encodings. Cryptology ePrint Archive, Report 2007/104, 2007, http://eprint.iacr.org/
12. http://www.cloakware.com
13. http://www.retrologic.com
14. http://www.yworks.com/en/products_yguard_about.htm

Appendix A: Proofs

Proof of Property 1: Let E be the set $\{\Delta \in \mathbb{F}_2^{64} \mid D_\Delta F(\mathbb{F}_2^{64}) \subset \overline{K}_k\}$.

- Let Δ be an element belonging to B_k. Let X be an element belonging to $\mathbb{F}_2^{64}$.

$$D_\Delta F(X) = (D_\Delta(b_1 \circ \pi_1 \circ M_1 \circ M_0(X)), \ldots, D_\Delta(b_{24} \circ \pi_{24} \circ M_1 \circ M_0(X)))$$

  According to the definitions, if $\Delta \in B_k$ then $M_0(\Delta) \in \mathcal{B}_k$ or equivalently $\pi_k \circ M_1 \circ M_0(\Delta) = 0$. Writting $D_\Delta(b_k \circ \pi_k \circ M_1 \circ M_0(X))$ as (1), we have :

$$\begin{aligned}(1) &= b_k \circ \pi_k \circ M_1 \circ M_0(X) \oplus b_k \circ \pi_k \circ M_1 \circ M_0(X \oplus \Delta) \\ &= b_k \circ \pi_k \circ M_1 \circ M_0(X) \oplus b_k(\pi_k \circ M_1 \circ M_0(X) \oplus \pi_k \circ M_1 \circ M_0(\Delta)) \\ &= b_k \circ \pi_k \circ M_1 \circ M_0(X) \oplus b_k(\pi_k \circ M_1 \circ M_0(X) \oplus 0) \\ &= b_k \circ \pi_k \circ M_1 \circ M_0(X) \oplus b_k \circ \pi_k \circ M_1 \circ M_0(X) = 0\end{aligned}$$

  This means that $D_\Delta F(X)$ belongs to $\overline{K}_k$ or equivalently Δ belongs to E. We conclude that $B_k \subset E$.
- Let Δ be any element of E. According to the definition of E, we have in particular $D_\Delta(0) \in \overline{K}_k$. This means that

$$b_k(0) \oplus b_k \circ \pi_k \circ M_1 \circ M_0(\Delta) = 0 \ ,$$

  or equivalently

$$b_k(0) = b_k \circ \pi_k \circ M_1 \circ M_0(\Delta) \ .$$

  We deduce that $\pi_k \circ M_1 \circ M_0(\Delta) = 0$ because b_k is a bijection. According to the definitions, it means that $M_0(\Delta) \in \mathcal{B}_k$ or equivalently Δ belongs to B_k. Therefore $E \subset B_k$. We conclude that $E = B_k$.
- Note that in fact B_k is the kernel of $\pi_k \circ M_1 \circ M_0$. Since $\mathrm{rank}(\pi_k \circ M_1 \circ M_0)$ is less or equal to 4, and greater or equal to 1, we have simultaneously $60 \leq \dim(B_k) \leq 63$ and the probability that Δ belongs to B_k when Δ is randomly chosen, is equal to $\frac{\dim(B_k)}{2^{64}}$. The results follows.

$\square$

Proof of Lemma 1: First recall that $\mathcal{B}_k = \langle e_j \mid \pi_k \circ M_1(e_j) = 0\rangle$ and $\mathcal{E}_k = \langle e_j \mid \pi_k \circ M_1(e_j) \neq 0\rangle$. Let j and k be two distinct integers, then the following conditions are equivalent.

- $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$.
- $\pi_k \circ M_1(e_i) = 0$ or $\pi_j \circ M_1(e_i) = 0$ for any integer $i \in [1, 64]$.
- $\pi_k \circ M_1(X) = 0$ or $\pi_j \circ M_1(X) = 0$ for any vector $X \in \mathbb{F}_2^{64}$.

We conclude that if $X \in \mathcal{E}_j$ and $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$ then $\pi_k \circ M_1(X) = 0$ or equivalently $X \in \mathcal{B}_k$.

Consider $X \neq 0$ belonging to $\bigcap_{j \neq k} \mathcal{B}_j$. We have that $\pi_j \circ M_1(X) = 0$ for any $j \neq k$. Note that M_1 is injective. Therefore $M_1(X) \neq 0$ and $\pi_k \circ M_1(X) \neq 0$. We conclude that all the bits of $M_1(X)$ that touch b_j ($j \neq k$) are zeros. Therefore, for any non-zero component e_i of X, $M_1(e_i)$ touches b_k or equivalently $X \in \mathcal{E}_k$, and $\bigcap\limits_{j \neq k} \mathcal{B}_j \subset \mathcal{E}_k$.

Let us use an argument by contraposition. Consider $e_i \notin \bigcap\limits_{j \neq k} \mathcal{B}_j$. Then, there exists $j \neq k$, such that $e_i \notin \mathcal{B}_j$, i.e. $\pi_j \circ M_1(e_i) \neq 0$ or equivalently $e_i \in \mathcal{E}_j$. Therefore, according to the previous three equivalent conditions, $e_i \notin \mathcal{E}_k$. We deduce that for any $e_i \in \mathcal{E}_k$ we have $e_i \in \bigcap\limits_{j \neq k} \mathcal{B}_j$. It means that $\mathcal{E}_k = \langle e_i \mid e_i \in \mathcal{E}_k \rangle \subset \bigcap\limits_{j \neq k} \mathcal{B}_j$. We conclude $\mathcal{E}_k = \bigcap\limits_{j \neq k} \mathcal{B}_j$.

$\square$

Proof of Property 2: Let e_i be an element of $\mathcal{E}_k$ and j be an element of J_k. We have $\pi_k \circ M_1(e_i) \neq 0$ and $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$. It implies that $\pi_j \circ M_1(e_i) = 0$, and $e_i \in \mathcal{B}_j$. Therefore, $e_i \in \bigcap\limits_{j \in J_k} \mathcal{B}_j$, and $\langle e_i \mid e_i \in \mathcal{E}_k \rangle \subset \widehat{\mathcal{E}}_k$. $\square$

Proof of Property 3: We will first prove that $(\mathcal{B}_i \cap \mathcal{B}_j) \oplus \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle = \mathbb{F}_2^{64}$. Consider a canonical vector $e_k \notin \mathcal{B}_i \cap \mathcal{B}_j$. This is equivalent to $\pi_i \circ M_1(e_k) \neq 0$ or $\pi_j \circ M_1(e_k) \neq 0$. In other words $e_k \in \mathcal{E}_i$ or $e_k \in \mathcal{E}_j$, or equivalently $e_k \in \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$. This means that for any canonical vectors e_k of $\mathbb{F}_2^{64}$, we have either e_k belongs to $\mathcal{B}_i \cap \mathcal{B}_j$ or e_k belongs to $\langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$.

Assume that there exists a canonical vector $e_k \in (\mathcal{B}_i \cap \mathcal{B}_j) \cap \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$. We have $\pi_i \circ M_1(e_k) = \pi_j \circ M_1(e_k) = 0$, and either $\pi_i \circ M_1(e_k) \neq 0$ or $\pi_j \circ M_1(e_k) \neq 0$. It leads to a contradiction. Hence $(\mathcal{B}_i \cap \mathcal{B}_j) \cap \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$ contains no canonical vectors.

Assume now that there exists an element $\Delta \in (\mathcal{B}_i \cap \mathcal{B}_j) \cap \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$ having a non-zero component e_k. The vector Δ belongs to $(\mathcal{B}_i \cap \mathcal{B}_j)$, hence e_k belongs to $(\mathcal{B}_i \cap \mathcal{B}_j)$. Moreover Δ belongs to $\langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$, hence e_k belongs to $\langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$. Therefore e_k belongs to $(\mathcal{B}_i \cap \mathcal{B}_j) \cap \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle$ which is impossible. We conclude that $(\mathcal{B}_i \cap \mathcal{B}_j) \cap \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle = \{0\}$. Therefore $(\mathcal{B}_i \cap \mathcal{B}_j) \oplus \langle \mathcal{E}_i \cup \mathcal{E}_j \rangle = \mathbb{F}_2^{64}$.

We deduce that

$$
\begin{aligned}
64 &= \dim(\langle \mathcal{E}_i \cup \mathcal{E}_j \rangle) + \dim(\mathcal{B}_i \cap \mathcal{B}_j) \\
&= \dim(\mathcal{E}_i + \mathcal{E}_j) + \dim(\mathcal{B}_i \cap \mathcal{B}_j) \\
&= \dim(\mathcal{E}_i) + \dim(\mathcal{E}_j) - \dim(\mathcal{E}_i \cap \mathcal{E}_j) + \dim(\mathcal{B}_i \cap \mathcal{B}_j)
\end{aligned}
$$

Moreover $\mathcal{E}_i \oplus \mathcal{B}_i = \mathbb{F}_2^{64} = \mathcal{E}_j \oplus \mathcal{B}_j$. Hence $64 = 64 - dim(\mathcal{B}_i) + 64 - \dim(\mathcal{B}_j) - \dim(\mathcal{E}_i \cap \mathcal{E}_j) + \dim(\mathcal{B}_i \cap \mathcal{B}_j)$. The result follows. $\square$

Appendix B: Figures

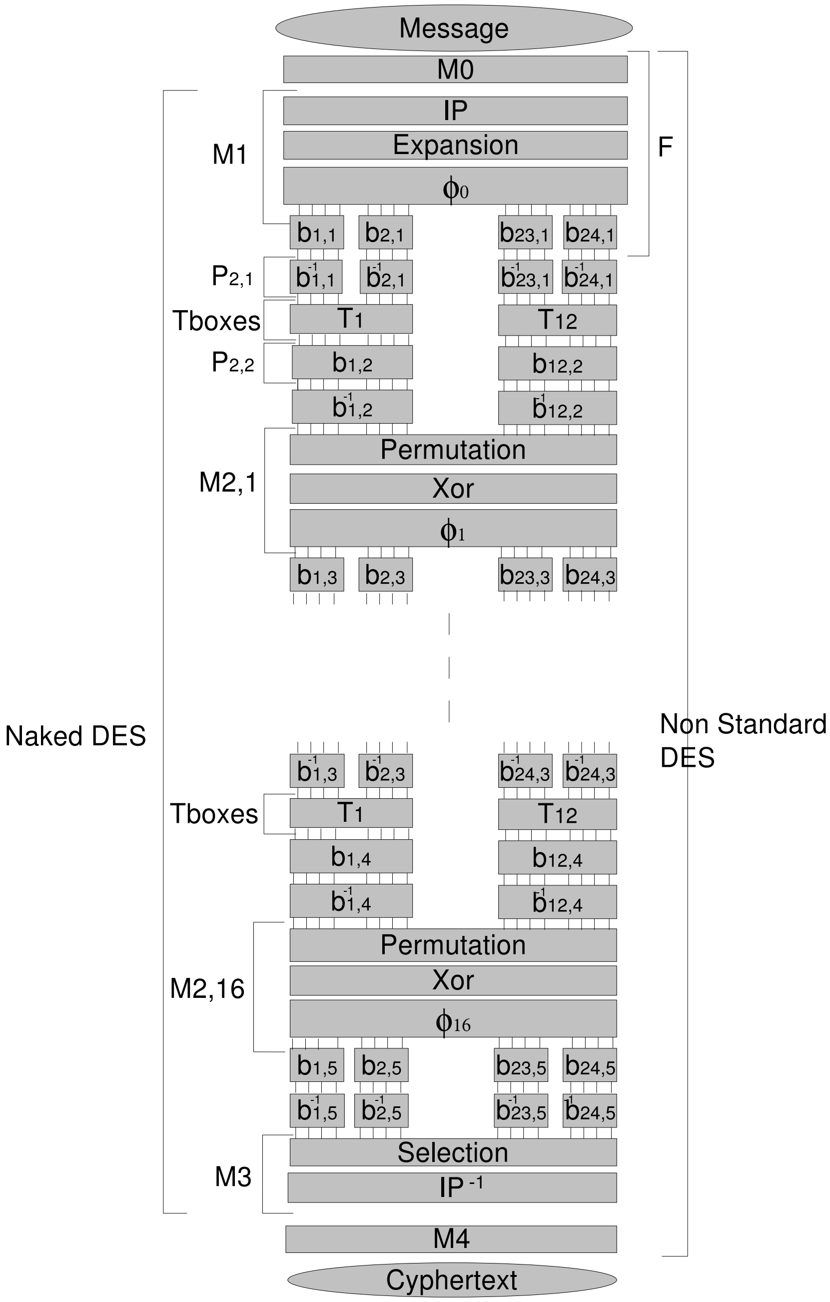

Fig. 7. "Naked-DES" and "Nonstandard-DES"

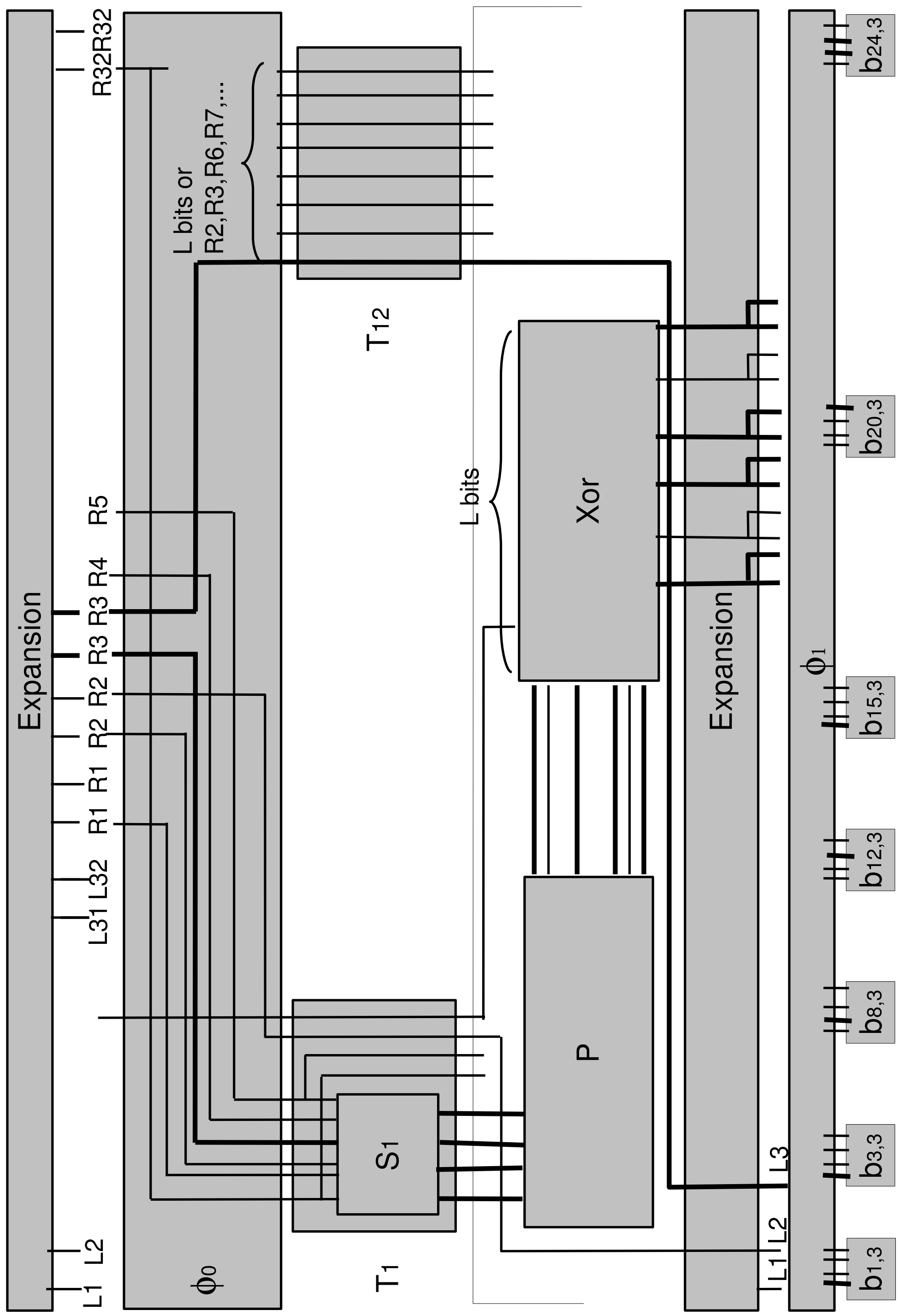

Fig. 8. General view of the attack