Cryptanalysis of white box DES implementations

Louis Goubin Jean-Michel Masereel Michaël Quisquater

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

Mars 19th, 2007

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 knowledge of the key. The goal of obfuscation of block ciphers is therefore to produce programs containing the key that could not be extracted by someone having access to the source code. This paper deals with the cryptanalysis of such methods of obfuscation, in particular for the DES. Such methods, called the "naked DES" and "non standard DES", were proposed by Chow *et al.* [4] in 2002. Some methods for the cryptanalysis of the "naked DES" were proposed by Chow *et al.* [4], Jacob *et al.* [5], and Link and Neuman [6]. In their paper, Link and Neuman [6] proposed an other 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).

1 Introduction

In 2002, Chow *et al.* [3, 4] suggested two different obfuscations, one for the AES, the other for the DES. The AES obfuscation was cryptanalysed by Billet *et al.* [1, 2] in 2004. Also Chow *et al.* [4] gived an attack on their first DES obfuscation version (called "naked DES"). Jacob *et al.* [5] and Link and Neuman [6], proposed two others attacks on the "naked DES". A second version of DES obfuscation, called "non standard DES", was given by Chow *et al.* [4]. This version seems not to have been cryptanalysed yet.

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, which was not cryptanalysed so far. Sectio n5 is devoted to our implementation of this attack. Finally, we conclude in Section 6.

2 DES obfuscation methods

Chow *et al.* [4] proposed two types of DES obfuscation. The first one, called "naked DES", produces a real DES. The second one, called the "non standard DES", is a slight modification of the standard DES algorithm. This last version is the one they recommend.

Let's describe the naked DES. The obfuscation starts with an affine function M_1, wich is the composition of the initial permutation and the expansion. This function is linear, so we can parsed it in a clever way, in many tables. Each round is then the concatenation of 12 T-boxes (derived from the S-boxes of the DES) followed by an affine function M_2 (derived from P and the xor operation). The last round of the obfuscation is followed by an affine function M_3 which is the final permutation. This function takes for arguments the outputs of the affine function M_2 of the last round and returns the cipher text. We will denote by A_i, one of these components (T-box or M_i).

Each components A_i are obfuscated between random non linear permutations P_1 and P_2, i.e. $P_1 \circ A_i \circ P_2$ (it is what Chow *et al.* [4] refers to io-block encoding). The resulting functions are stored in arrays in order to be used in the obfuscated program. Permutations P_1 and P_2 are chosen such that the composition of consecutive obfuscated components is the obfuscation of the composition of the components.

This obfuscation was cryptanalysed by the authors themselves [4]. In order to avoid this attack, they propose the "non standard DES". It consists in adding two affine bijections M_0 and M_4 in front and after the naked DES, respectively. It is not specified by Chow *et al.* [4] whether M_0 and M_4 are block encoded (i.e. respectively preceded and followed by non linear random permutations). In this paper, we consider that M_0 and M_4 are not block encoded.

On the other hand, Link and Neumann [6] improved the attack on the "naked DES", and suggested another solution which consists in merging the T-boxes and the affine function M_2 of each round. This way, we do not have access to the T-boxes outputs. Moreover, the M_2 of the different rounds are block encoded in an other way.

As far as we know, no one has published an attack neither on the "non standard DES", nor on the improved "naked DES". We adress this issue in this paper.

3 Attack on the naked DES

3.1 The principle

As mentioned before, the naked DES proposed by Chow *et al.* [4] was already cryptanalysed in the papers [4, 5, 6]. In this section, we show how to cryptanalyse the improved version of the naked DES proposed by Link and Neumann [6]. Note that our method works also for the naked DES proposed by Chow *et al.* [4]. In what follows, we will denote by "regular DES", the one described in the standard [8] (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 throught IP and the first round of a regular DES, are equal (for a given key). 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.

Let's go into the details. Remember that $f(., k)$ denotes the function of the regular DES, we will also denote it as $f_k(.)$. (L_0, R_0) denotes the image of the initial message 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 $f(x_0) \oplus f(X_0 \oplus \Delta)$ will be denoted by $D_\Delta f(X)$. Let's 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 ourself to $i = 1$). Therefore we will 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 bits index 2 and 3.
2. $L'_0 = L_0 \oplus D_\Delta f_k(R_0)$

Observe that the bits of R_0 index 2 and 3 only affect the output of S_1. Therefore, $f(R_0, k)$ and $f(R'_0, k)$ are identical except for the 4 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. 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 of the first round, and we will be able to distinguish this situation from the correct key guess.

Here is an overview of the attack:

- Choose a message X randomly.

- Compute $(L_0, R_0) = IP(X)$ with IP public.
- Choose Δ such that only bits index 2 and 3 are different from 0.
- For all possible 6 bits of round subkey k:
  - 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.
  - If this number is strictly greater than 2, then reject the 6-bit-subkey, else the 6-bit-subkey are probably correct.

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

3.2 Efficiency

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 produces 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 S1 (the bits index 32,1,...,5). Actually we can choose differents 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. 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 "non standard DES"

This section is dedicated to an attack on the "non standard DES". Remind that the "non standard DES" is a "naked DES" where the affine function M_1 is replaced by $M_1 \circ M_0$, where M_0 is a mixing bijection (see Chow *et al.* [4]). As mentionned 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 4×4 bijections (the same principle applies for the obfuscation proposed by Link and Neuman [6] where bijections are from 8 to 8 bits). Moreover, we assume that the T-Boxes follow the same ordering in the differents rounds. In what follows, we will not consider IP (inside M_1) for the sake of clarity. It does not change anything to the argument.

Denote by $F : \mathbb{F}_2^{64} \to \mathbb{F}_2^{96}$ the obfuscation of $M_1 \circ M_0$. We summarize the situation in the figures below. The function $\phi : \mathbb{F}_2^{96} \to \mathbb{F}_2^{96}$ is a bit-permutation (48 positions are determined by the regular DES operation and the others 48 bits are chosen randomly).

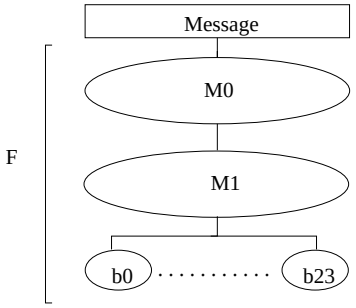

Figure 1: F function

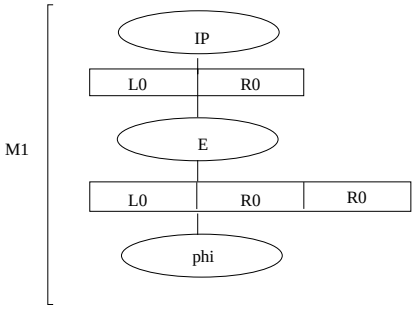

Figure 2: M_1 function

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 "non standard DES" is based on the one on the "naked DES". Our approach is based on a differential truncated 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 to provide gradually informations on the key and the matrix M_0^{-1}. Full key and the matrix M_0^{-1} are known at the end. The way we store information on 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* list. 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 in shortening the lists using some tricks. Our method can therefore be understood as a "filtering process". The different filters are described below.
Section 4.1 describes a preliminary step almost independant of the structure of the block cipher. It consists in finding vector spaces associated to a particular io-block encoding bijection at the output of F. This step allows to get global information on M_0^{-1}.
Section 4.2 describes a set of filters intending to refine information on M_0^{-1}. These steps are highly related to the studied block cipher. The first filter, described in Section 4.2.1, allows to distinguish lists that are associated to canonical vectors belonging either to right bits or left bits of the input of the first round. The second filter, described in Section 4.2.2, extracts all the lists (marked as "right" in the previous filter) touching a single S-box. The third filter, described in Section 4.2.3, gathers the lists (marked as "left" in the previous filter) in sets associated to the output of S-boxes. Section 4.2.4 links T-Boxes (obfuscation of the keyed S-boxes) to S-Boxes. This information allows the last filter, presented in Section 4.2.5, to specify precisely 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 Bloc level analysis of $M_1 \circ M_0$

Denote by K_k the space $(\{0\}^{4k} \times \mathbb{F}_2^4 \times \{0\}^{92-4k})$, and by $\overline{K}_k$, the space $(\mathbb{F}_2^{4k} \times \{0\}^4 \times \mathbb{F}_2^{92-4k})$. 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 their vectors influence only one io-block encoding bijection at the output of $M_1 \circ M_0$. This would allow to act specificaly on 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 achieve. 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.
Let X a vector in $\mathbb{F}_2^{96}$, let k be an integer, $k \in [0, 23]$, π_k denotes the projection $\pi_k : (\mathbb{F}_2^4)^{24} \to \mathbb{F}_2^4 : X = (x_1, \ldots, x_{24}) \mapsto x_k$. Let b_k be the $(k+1)th$ io-block encoding bijection at the output of $M_1 \circ M_0$. The function F is written as

$$F(X) = (b_0 \circ \pi_0 \circ M_1 \circ M_0(X), b_1 \circ \pi_1 \circ M_1 \circ M_0(X), \ldots, b_{23} \circ \pi_{23} \circ M_1 \circ M_0(X))$$

Definition 4.1 *Let k be an integer, $k \in [0, 23]$. 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 4.2 *Let k be an integer, $k \in [0, 23]$. 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. $\mathcal{E}_k = \langle e_j \mid \pi_k \circ M_1(e_j) \neq 0 \rangle$.*

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

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

Property 4.3 *For all k integer, $k \in [0, 23]$, $B_k = \{\Delta \in \mathbb{F}_2^{64} \mid D_\Delta F(\mathbb{F}_2^{64}) \subset \overline{K}_k\}$, 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 $60 \leq \dim(B_k) < 64$.*

Proof: 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 $\mathbb{F}_2^{64}$.

  $$D_\Delta F(X) = (D_\Delta(b_0 \circ \pi_0 \circ M_1 \circ M_0(X)), \ldots, D_\Delta(b_{23} \circ \pi_{23} \circ M_1 \circ M_0(X)))$$

  According to the definitions, if $\Delta \in B_k$ then $M_0(\Delta) \in \mathcal{B}_k$ or equivalently $\pi_k \circ M_1 \circ M_0(\Delta) = 0$. Let's compute $D_\Delta(b_k \circ \pi_k \circ M_1 \circ M_0(X)) = (1)$.

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

  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 $\text{rank}(\pi_k \circ M_1 \circ M_0)$ 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.

□

Combining Definition 4.2 and Property 4.3, 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 less than four non-zero components e_i, all of them touching the $(k+1)th$ io-block encoding bijection through M_1. Due to the Property 4.3, it is easier to recover a basis for B_k's, than for E_k's. That's why we will first recover all the B_k's. Using Property 4.3, 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.
Let's now explain how to recover $\widehat{E}_k$. First, let's remark that for all $X \in \mathbb{F}_2^{64}$ and for all $\Delta \in \mathbb{F}_2^{64}$, 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's introduce the following lemma.

Lemma 4.4 *Let k be an integer belonging to $[0, 23]$. If $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$ for all integer j distinct from k belonging to $[0, 23]$, then*

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

Proof: The proof is available in the appendix.

□

Since M_0 is a bijection, this lemma means that if $\mathcal{E}_j \cap \mathcal{E}_k = \{0\}$ for all integer $j \in [0, 23]$ different from k, then $E_k = \bigcap_{j \neq k} B_j$. Nevertheless, due to the bit-duplication, there exist indexes 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 $[0, 23]$.

Property 4.5 *For all integer* $k \in [0, 23]$, $\mathcal{E}_k \subseteq \widehat{\mathcal{E}}_k$.

Proof: The proof is available in the appendix.

□

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

Property 4.6 *For all integer* $i \in [0, 23]$ *and for all integer* $j \in [0, 23]$

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

Proof: The proof is available in the appendix.

□

A straight forward applicationof this property to the definition of J_k leads to $J_k = \{j \in [0, 23] \mid 64 = \dim(B_j) + \dim(B_i) - \dim(B_i \cap B_j)\}$. 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 searche. 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 specified io-block encoding bijection. It allowed us to get some information on M_0^{-1}. In this section, we refine our search and this will allow us to get enough information on M_0^{-1} in order to apply our method on the "naked DES" to "non standard DES". Our algorithm works sequentially and consists in a "filtering process". The different filters are described below.

4.2.1 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. 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 distincts io-block encoding bijection 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. By considering intersections between the spaces $\widehat{E}_k$, 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 actually 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.

4.2.2 Recovering middle bits

In order to apply our attack presented in Section 3, we need to know exactly the preimage of canonical vectors touching only a single S-Box of the first round (e.g. Right bits index 2,3,6,7,10,...). In what follows, we will refer to such a canonical vector as a *middle bit*.
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 actual value for the other T-Boxes. 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 pointed by several canonical vectors can be divided into two shorter lists; one list is pointed by middle bits while the other is pointed by non-middle bits.

Remarks: 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.

4.2.3 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. 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 $b_i \in BS$ we have $D_{\Delta_m \oplus \Delta_l} b_i(X_0) = 0$, 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).

4.2.4 Chaining

In this section, we will try to determine precisely the correspondance between T-boxes and S-boxes. Due to the remark in Section 4.2.2, we know which are the T-boxes containing an S-box. The probability that a selected T-box, denoted by $T1$, contains $S1$ 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 $T1$. Selecting one of these T-Boxes randomly, the probability that it contains $S2$ 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. Note that the probability to determine the right correspondance is $1/8 \times 1/2 = 1/16$.

4.2.5 Bits positions

At this stage, we have recovered between others, 32 preimages corresponding to unspecified left canonical vectors. In order to determine the correspondance, 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 to check whether the image of this vector in front of the second round is a middle bit (cf. 4.2.2). Assuming that the T-Boxes follow the same ordering in the differents rounds, preimages corresponding to a middle bit (resp. non-middle bit) can be distinguished observing the indexes of the touched T-Boxes.
For example, for the first S-box, among the 4 identified left canonical vectors preimages,

- the one that is the preimage of a middle bit of S_6 (resp. $S8$) in the second round is the preimage of e_{23} (resp. e_{31}).
- the one that is not the preimage of a middle bit and is in the input of S_2 and S_3 (resp. $S4$ and $S5$) of the second round, is the preimage of e_9 (resp. e_{17}).

4.3 The attack

In Section 4.2, we have shown how to recover all the preimage 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 are pointed by some canonical vectors (their number is the dimension of the vector space spanned by the list). In this case, we select linearly

independant vectors in the list and we associate each of them to the canonical vector pointing to the list. Therefore, we are in the context of the attack of the naked-DES modulo 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 provide 48 key-bit candidates. Note that it may happen that wrong keys will be recovered. More importantly, it may happen that no key exists 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 all 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 "non standard 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 key-bit candidate and a candidate for M_0^{-1}. We make an exhaustive search in order to determine the 8 remaining bits. 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 key-bit candidate is wrong. If all the 8 key-bit candidate are wrong, we discard this particular M_0^{-1}. Note that this method also works if M_4 has a io-block encoding bijections at the 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 non-standard DES) can be applied on the Link and Neumann [6] 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 for the key, and for M_0^{-1} decrease, leading to a unique 48 key-bit candidate and a unique M_0^{-1} candidate. We have tested our attack on thousands

of obfuscated implementations of DES (both "naked" and "non-standard" DES). The figure 3 shows the time needed to complete the attack. We can observe that 95% of the attacks requires less than 50 secondes, and 75% less than 17 secondes. The mean time is about 17 secondes. However, the attacks were executed on a standard PC, and the code was not optimized, so we can easily divise the time by a factor 4.

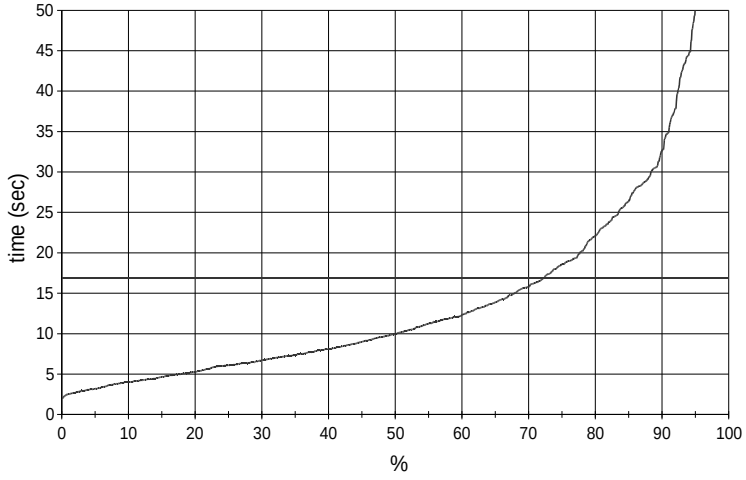

Figure 3: Repartition of the attacks durations

6 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 implemented our method with a C code and applied it successfully to more thousands of obfuscated implementations of DES (both "naked" and "non standard" DES). So far, all the studied instances lead to a unique candidate for the DES key and similarly for the M_0 and M_4 secret linear transformations. The key and the two linear transforms are obtain within 17 seconds on average.

References

[1] O. Billet. *Cryptologie Multivariable* Ph.D. thesis University of Versailles, December 2005.

[2] 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.

[3] 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*, volume 2595 of *LNCS*, pages 250–270. Springer-Verlag, 2002.

[4] 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 2002*, volume 2595 of *LNCS*, pages 1–15. Springer-Verlag, 2002.

[5] 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.

[6] H.E. Link and W.D. Neumann. Clarifying obfuscation: Improving the security of white-box encoding. 2004. http://eprint.iacr.org/.

[7] J. Patarin and L. Goubin. Asymmetric cryptography with S-boxes. In *Proc. 1st International Information and Communications Security Conference*, pages 369–380, 1997.

[8] http://www.itl.nist.gov/fipspubs/fip46-2.htm

Appendix: Proofs

Proof of Lemma 4.4: 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 all integer $i \in [1, 64]$.
- $\pi_k \circ M_1(X) = 0$ or $\pi_j \circ M_1(X) = 0$.for all 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 all $j \neq k$. 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 $M_1 \circ (X)$ 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 $\bigcap_{j \neq k} \mathcal{B}_j \subset \mathcal{E}_k$.

Let's use an argument by contraposition. Consider $e_i \notin \bigcap_{j \neq k} \mathcal{B}_j$. 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 all $e_i \in \mathcal{E}_k$ we have $e_i \in \bigcap_{j \neq k} \mathcal{B}_j$. It means that $\mathcal{E}_k = \langle e_i \mid e_i \in \mathcal{E}_k \rangle \subset \bigcap_{j \neq k} \mathcal{B}_j$. We conclude $\mathcal{E}_k = \bigcap_{j \neq k} \mathcal{B}_j$.

□

Proof of Property 4.5: 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 $e_i \in \mathcal{B}_j$. Therefor, $e_i \in \bigcap_{j \in J_k} \mathcal{B}_j$, and $\langle e_i \mid e_i \in \mathcal{E}_k \rangle \subset \widehat{\mathcal{E}_k}$.

□

Proof of Property 4.6: 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 all canonical vector 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.

□