An FPGA Implementation of the Linear Cryptanalysis

Francois Koeune, Gael Rouvroy, Francois-Xavier Standaert,
Jean-Jacques Quisquater, Jean-Pierre David, Jean-Didier Legat
{koeune,rouvroy,standaert,quisquater,david,legat}@dice.ucl.ac.be

UCL Crypto Group
Place du Levant, 3, B-1348 Louvain-La-Neuve, Belgium

Abstract. This paper deals with cryptographic concepts. It presents a hardware FPGA implementation of linear cryptanalysis of DES[1]. Linear cryptanalysis is the best attack known able to break DES faster than exhaustive search. Matsui's original attack [4, 5] could not be applied as such, and we had to implement a modified attack [1] to face hardware constraints. The resulting attack is less efficient than Matsui's attack, but fits in our hardware and breaks a DES key in 12-15 hours on one single FPGA, therefore becoming the first practical implementation to our knowledge. As a comparison, the fastest implementation known so far used the idle time of 18 Intel Pentium III MMX, and broke a DES key in 4.32 days.
Our fast implementation made it possible for us to perform practical tests, allowing a comparison with theoretical estimations.

Keywords: Cryptography, linear cryptanalysis, FPGA, DES.

1 Introduction

Linear cryptanalysis [1, 4, 5] is a cryptanalytic technique that takes advantage of possible input-output correlations over a cipher. Evaluating this relationship for an sufficient number of plaintext/ciphertext pairs (typically 2^{43}, for a full DES), it is possible to recover some bits of the key faster than an exhaustive search.

Although linear cryptanalysis is the best attack known against DES nowadays, this attack still has a "theoretical" flavour, in the sense that very few experimental applications have actually been performed: a single experimentation for a full DES cipher has been performed in [5], and, until recently, remained the only practical test to our knowledge.

However, recent technological advances have made the required computing power reachable, as is witnessed by a set of 21 experiments [2, 3], using the idle time of 18 Intel Pentium III MMX, capable of performing an attack in 4.32 days.

This paper proposes an FPGA implementation completing the attack in 12-15 hours, using hardware roughly worth $3500. We believe that our implementation

[1] DES : Data Encryption Standard, the old U.S. cipher standard

* This work has been funded by the Walloon region (Belgium) through the research project TACTILS.

is the fastest implementation known. Due to tight memory constraint, Matsui's original attack could not be implemented as such. Therefore we implement a variant of it ([1]), which turns out to be less efficient on a theoretical point of view, but gave birth to a very fast implementation. In fact, this attack can be considered as more efficient than Matsui's, in the sense that it requires less plaintext-ciphertext (2^{42} vs. 2^{43}) pairs, but recovers only 7 (resp. 14, by using the dual equation) key bits rather than 13 (resp. 26).

The paper is organized as follows: section 2 describes our FPGA's main characteristics; section 3 reminds the basic principles of linear cryptanalysis; section 4 presents the modified attack and its expected theoritical efficiency; section 5 discusses the attack completion (i.e. recovering the full key); finally, section 6 summarizes the results we obtained on a set of 27 practical tests.

2 Hardware Ressources

All our experiments were carried out on a Virtex1000BG560-4 FPGA board developed by DICE [2]. The board is composed of a control FPGA (FLEX 10K) and a VIRTEX1000 FPGA[3] associated with several processors (ARM and PIC) and fast access memories. The board has multiple compatible PC interfaces (PCI, RS232,USB). To carry out our simulations we used a PCI communication.

3 Linear cryptanalysis

This section is a brief reminder of Matsui's linear cryptanalysis [4, 5]. This attack is based on the existence of some unbalanced linear relationship between input and output of a reduced-round version of the target encryption scheme. In the case of DES, Matsui used the relationship

$$P_L[15] \oplus P_H[7,18,24,29] \oplus C_L[7,18,24] = K_1[22] \oplus K_3[22] \oplus K_4[44] \\ \oplus K_5[22] \oplus K_7[22] \oplus K_8[44] \oplus K_9[22] \oplus K_{11}[22] \oplus K_{12}[44] \oplus K_{13}[22]. \quad (1)$$

Basically, this relationship means that the exclusive-or of some well-chosen bits of the plaintext (namely, the 7th, 18th, 24th, 29th bits of its high-order part) and some well-chosen bits of the ciphertext is equal to the exclusive-or of some well-chosen bits of the key with probability different from $\frac{1}{2}$.

We can easily calculate its dual, obtained by reversing the expression

$$P_L[7,18,24] \oplus C_L[15] \oplus C_H[7,18,24,29] = K_2[22] \oplus K_3[44] \oplus K_4[22] \\ \oplus K_6[22] \oplus K_7[44] \oplus K_8[22] \oplus K_{10}[22] \oplus K_{11}[44] \oplus K_{12}[22] \oplus K_{14}[22], \quad (2)$$

where $X[7,18,24] := X[7] \oplus X[18] \oplus X[24]$. Those characteristics are the best linear approximations of 14-round DES cipher. They are satisfied with probability $p = \frac{1}{2} - 1.19 \times 2^{-21}$.

[2] UCL Microelectronics laboratory (see `http://www.dice.ucl.ac.be`).
[3] This FPGA counts about 6144 CLB's.

Expression[4] (1) is then extended to the full 16 rounds by adding two non-linear round functions respectively in the first and 16-round:

$$\begin{aligned} P_L[7,18,24,29] \oplus P_H[15] \oplus F_1(P_L,K_1)[15] \oplus C_H[7,18,24] \oplus \\ F_{16}(C_L,K_{16})[7,18,24] = K_2[22] \oplus K_4[22] \oplus K_5[44] \oplus K_6[22] \oplus \\ K_8[22] \oplus K_9[44] \oplus K_{10}[22] \oplus K_{12}[22] \oplus K_{13}[44] \oplus K_{14}[22], \end{aligned} \tag{3}$$

where $F_1(P_L, K_1)$ denotes the first round function. This relationship keeps exactly the same probability as eq. (1). In fact only 6 bits of K_1 (resp. K_{16}) influence the value of $F_1(P_L, K_1)[15]$ (resp. $F_{16}(C_L, K_{16})[7, 18, 24]$).

If we compute this equation for all 4096 possibilities of the key K_1 and K_{16} a large number of plaintexts, knowing that only one of these 4096 keys is correct, we will find one significative probability corresponding to the 12 correct key bits. The following algorithm summarizes this idea:

Algorithm

1. For each candidate K^i (i=1,2,... 4096) of (K_1,K_{16}), let T_i be the number of plaintexts such that the left side of the eq. (3) is equal to zero.
2. Let T_{max} be the maximal value, T_{min} the minimal value of all T_i's and N the number of plaintexts/ciphertexts.
   If $|T_{max} - \frac{N}{2}| > |T_{min} - \frac{N}{2}|$, then adopt the key candidate corresponding to T_{max}.
   If $|T_{max} - \frac{N}{2}| < |T_{min} - \frac{N}{2}|$, then adopt the key candidate corresponding to T_{min}.

An extra bit can be found thanks to relation (3). Indeed, as K_1 and K_{16} were found thanks to the precedent algorithm, we can derive the value of $K_2[22] \oplus K_4[22] \oplus K_5[44] \oplus K_6[22] \oplus K_8[22] \oplus K_9[44] \oplus K_{10}[22] \oplus K_{12}[22] \oplus K_{13}[44] \oplus K_{14}[22]$. It is therefore possible to recover 12+1 bits of the key. The same treatment can be applied to the dual equation (3), thus yielding a total of 26 bits. The remaining 30 unknown key bits have to be searched exhaustively.

4 A chosen-plaintext linear cryptanalysis

As described in [5], Matsui's Linear Cryptanalysis allows to find these 26 key bits with 2^{43} known-plaintext. Nevertheless, for a hardware implementation, the main problem of this attack is the 2×2^{12} counters (43-bits wide) needed to perform the key guess. Knowing that we have about 24000 LUT's on our FPGA, the implementation of 2^{12} parallelized counters is much too expensive to be realistic (more than 350000 LUT's).

So, we have to reduce the number of needed counters. Looking back at equation (3)[5], we see that these counters are induced by the terms $F_1(P_L, K_1)[15]$

[4] We will leave the second relationship aside in this discussion, since it is the first one's dual.

[5] The dual equation can be treated similarly.

and $F_{16}(C_L, K_{16})[7, 18, 24]$, each of which depends on 6 key bits. If we could force one of these terms (say, F_1) to a constant value, we would get rid of 2^6 counters.

Due to the non-linear character of F_1 (remember that F_1 – or at least the parts of its output we are interested in – basically corresponds to the S-box S_5), the only way to force $F_1(P_L, K_1)[15]$ to a constant value seems to be to fix its input. As the key bits are obviously constant, all we have to do is to fix 6 bits of P_L to a constant value. As a consequence, this attack becomes a chosen-plaintext attack. It is an attack proposed in [1].

Let us have a look at the success rate of this algorithm. In [4], the following lemmas are proposed:

Lemma 1. *Let N be the number of given random plaintexts and p be the probability that equation 3 holds, and assume $|p - \frac{1}{2}|$ is sufficiently small. Then, the success rate of the algorithm depends on $l_1, l_2, \ldots, l_d$ (as defined in Lemma 2), and $\sqrt{N}|p - \frac{1}{2}|$ only.*

Generally speaking, it is not easy to calculate numerically the accurate probability above. However, under a condition it can be possible as follows.

Lemma 2. *With the same hypotheses as Lemma 1, let $q^{(i)}$ be the probability that the following equation holds for a subkey $K_n^{(i)}$ and a random variable X:*

$$F_n(X, K_n)[l_1, l_2, \ldots, l_d] = F_n(X, K_n^{(i)})[l_1, l_2, \ldots, l_d] \tag{4}$$

Then if $q^{(i)}$'s are independent, the success rate of the algorithm is

$$\int_{x=-2\sqrt{N}|p-\frac{1}{2}|}^{\infty} \Big(\prod_{K_n^{(i)} \neq K_n} \int_{-x-4\sqrt{N}(p-\frac{1}{2})q^{(i)}}^{x+4\sqrt{N}(p-\frac{1}{2})(1-q^{(i)})} \frac{1}{\sqrt{2\Pi}} e^{\frac{-y^2}{2}} dy \Big) \frac{1}{\sqrt{2\Pi}} e^{\frac{-x^2}{2}} dx \tag{5}$$

where the product is taken over all subkey candidates except K_n.

In the case we are considering, we have $d = 3$ and $l_1 = 7, l_2 = 18, l_3 = 24$. Then a numerical calculation of expression (5) is as follows.

N	$\lvert p-\frac{1}{2}\rvert^{-2}$	$2\lvert p-\frac{1}{2}\rvert^{-2}$	$4\lvert p-\frac{1}{2}\rvert^{-2}$	$8\lvert p-\frac{1}{2}\rvert^{-2}$	$16\lvert p-\frac{1}{2}\rvert^{-2}$
Success rate	20.1%	37.8%	64.1%	88.8%	98.8%

Table 1. Success rate

In addition we can write this table with $p = \frac{1}{2} - 1.19 \times 2^{-21}$:

N	2^{37}	2^{38}	2^{39}	2^{40}	2^{41}	2^{42}	2^{43}	2^{44}	2^{45}	2^{46}
Success rate	2.1%	3.0%	4.6%	8.0%	14.6%	27.8%	50.2%	77.8%	95.7%	99.8%

Table 2. Success rate with $p = \frac{1}{2} - 1.19 \times 2^{-21}$

It is of course difficult to compare this to Matsui's results, since the latter recovers 13 bits[6] rather than 6+1. As a comparison basis, we nevertheless used equation (5) to show the theoretical success probability of Matsui's 14-round attack[7], which is as follows:

N	$\lvert p - \frac{1}{2}\rvert^{-2}$	$2\lvert p - \frac{1}{2}\rvert^{-2}$	$4\lvert p - \frac{1}{2}\rvert^{-2}$	$8\lvert p - \frac{1}{2}\rvert^{-2}$	$16\lvert p - \frac{1}{2}\rvert^{-2}$
Success rate	4.8%	17.7%	51.3%	87.0%	98.8%

Table 3. Success rate in Matsui's case

N	2^{37}	2^{38}	2^{39}	2^{40}	2^{41}	2^{42}	2^{43}	2^{44}	2^{45}	2^{46}
Success rate	0.1%	0.1%	0.3%	0.8%	2.5%	9.3%	31.9%	71.8%	95.3%	99.8%

Table 4. Success rate in Matsui's case with $p = \frac{1}{2} - 1.19 \times 2^{-21}$

Therefore we can see that this method needs less plaintext/ciphertext pairs than Matsui's one, but retrieves less key bits. The next section discusses the implication this has on a complete attack.

5 Completing the attack

Once these 6+1 bits of information have been obtained, the question becomes of course: "how can they be exploited to obtain the complete key".

Since the 6 bits yielded by the previous attack belong to the 12 involved in the classical form of the linear cryptanalysis, we can now set up a classical attack[8], with these 6 bits fixed to the values we found in preceding phase. This would give the 6 other bits and only require 2^6 counters to maintain, which is clearly achievable by the FPGA. Applying the same treatment to the dual equation (eq. (3)) would provide us with a total of 26 key bits.

[6] The 13 other bits recovered by Matsui's attack are obtained by using the dual characteristic; as will be shown in next section, the same can be done in our case.

[7] Due to the large (4095) number of factors involved, the equation could not be computed exactly; therefore we used an approximation.

[8] A known plaintext attack.

Let us consider the success rate of this method. The characteristic we use in the second phase has probability $\frac{1}{2} - 1.19 \times 2^{-21}$. Using equation (5), we obtain the following values[9]:

N	2^{37}	2^{38}	2^{39}	2^{40}	2^{41}	2^{42}	2^{43}	2^{44}	2^{45}	2^{46}
Success rate	2.1%	3.1%	5.0%	8.9%	17.6%	35.4%	63.9%	90.1%	99.3%	100%

Table 5. Success rate to complete the attack with $p = \frac{1}{2} - 1.19 \times 2^{-21}$

We obtain the success probability of the global attack by multiplying the probabilities of the first and second phase (tables 2 and 5), as summarized in table 6. This is to be compared with Matsui's success rate, given by table 4.

N	2^{37}	2^{38}	2^{39}	2^{40}	2^{41}	2^{42}	2^{43}	2^{44}	2^{45}	2^{46}
Success rate	0.0%	0.1%	0.2%	0.7%	2.6%	9.8%	32.1%	70.1%	95.0%	99.8%

Table 6. Success rate of global attack

It turns out that these two tables are very close one to the other. One could thus be tempted to conclude that this two-phase attack is as efficient as Matsui's. Unfortunately, it is not possible to reuse for second phase the plaintext/ciphertext pairs used in first phase. As these pairs were explicitly constructed to make the first round constant, they cannot teach us any information about the corresponding key bits.

Consequently, this attack finally requires twice the amount of plaintext/ciphertext pairs required by Matsui for comparable efficiency. Its only advantage (besides the fact that 6 bits of information are already available at mid-course) is that it fits in our FPGA, and can actually be carried out in roughly 12 hours (as a comparison, the only actual implementation of linear cryptanalysis that we know [2, 3] performs an attack in 4.32 days. An exhaustive search of the remaining 30 bits would take about 3 seconds.

Remark: The above estimations only take into account the probability for the right key to be the *first* one in our guess list. In fact, a more efficient method is used in [5]: the candidates are sorted according to their ranking in the linear estimation, and are successively used as basis for exhaustive search. Similarly, we could of course simultaneously treat the 2^t more likely candidates yielded by first phase, combining them with all possible values for the 6 remaining bits. In view of the place left by our implementation on the Xilinx FPGA, it appears we could set $t = 3$. Corresponding success probabilities are difficult to derive theoretically (Matsui's estimations were obtained by extrapolating the results obtained against an 8-round DES). The next section summarizes the results we obtained by actually running the attack.

[9] These value are not the same as table 4, since the considered S-box (and thus $d, l_1, \ldots l_i$) is different.

6 Experimental results

In this section, we give a description of the results we got running the first phase of the attack (recover 6 key bits) on one single Xilinx FPGA. We carried out the experiments at a work frequency = 66.6 MHz ($=2^{26}$) and we parallelized 6 attack blocks. Therefore we are able to compute 6×2^{26} equations per second. So, 2^{43} evaluations take about 6 hours.

We carried out tests with 27 different keys. Table 7 summarizes the experimental success rate to rank first the right subkey candidate for various amounts N of chosen-plaintext/ciphertext pairs:

N	2^{36}	2^{37}	2^{38}	2^{39}	2^{40}	2^{41}	2^{42}	2^{43}
Success rate	0%	0%	0%	4%	4%	19%	37%	70%

Table 7. Experimental success rate in the first phase

These experimental results probably suggest that Matsui's theoretical analysis is slightly pessimistic (see table 2). Our hardware design could help to accurate existing mathematical model.

7 Conclusion

This paper presented the first known FPGA implementation of linear cryptanalysis. Due to FPGA constraints, we choose an adapted attack that makes it less memory-consuming. The resulting attack is less efficient (by a factor 2) than the original one, but can actually be deployed on reasonably expensive hardware and is capable of breaking a full DES key in 12-15 hours, including final exhaustive search. In addition, it is worth noting that with the new Xilinx FPGA[10], we would be able to carry out the same attack in about 1 hour, without changing the HDL code. Therefore, in some applications, FPGA's can be used as powerful cryptographic calculation tools.

References

1. L.R. Knudsen and J.E. Mathiassen A Chosen-Plaintext Linear Attack on DES. In Bruce Schneier, editor, *Proc. of FSE'00*, LNCS, pages 262–272. Springer, 2000.
2. P. Junod. Linear cryptanalysis of DES. Master's thesis, Swiss Institute of Technology, Zurich, 2000.
3. P. Junod. On the complexity of Matsui's attack. In *Proc. of SAC'01*, LNCS, pages 216–230. Springer, 2001.
4. M. Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology - EuroCrypt '93*, pages 386–397, Berlin, 1993. Springer-Verlag. Lecture Notes in Computer Science Volume 765.

[10] Xilinx VIRTEX-II XC2V8000.

5. M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Yvo Desmedt, editor, *Advances in Cryptology - Crypto '94*, pages 1–11, Berlin, 1994. Springer-Verlag. Lecture Notes in Computer Science Volume 839.
6. J.M. Rabaey. *Digital Integrated Circuits*. Prentice Hall, 1996.
7. Xilinx. Virtex 2.5V field programmable gate arrays data sheet. available from `http://www.xilinx.com`.