Enhancing Collision Attacks

Hervé Ledig, Frédéric Muller, and Frédéric Valette

DCSSI Crypto Lab 51, Boulevard de Latour-Maubourg
75700 Paris 07 SP France
{Frederic.Muller, Frederic.Valette}@sgdn.pm.gouv.fr

Abstract. Side Channel Attacks (SCA) have received a huge interest in the last 5 years. These new methods consider non-cryptographic sources of information (like timing or power consumption) in addition to traditional techniques. Consequently block ciphers must now resist a variety of SCAs, among which figures the class of "collision attacks". This recent technique combines side channel information with tools originally developed for block cipher or hash function cryptanalysis, like differential cryptanalysis for instance.
In this paper, we propose techniques to enhance collision attacks. First we describe a general framework for collision attacks against Feistel ciphers that extends and improves on previous results specifically obtained against DES. Then, we describe an improved method to attack DES using "almost collisions". Indeed we observed that taking into account internal states which are abnormally similar results in more efficient attacks. Some experimental results obtained against a DES implementation are finally presented.

1 Introduction

The idea of using side channel information to break cryptosystems implemented on a tamper-resistant device (typically think of this device as a smart-card) appeared in 1996 following the initial work by Kocher [6, 7]. This new class of attacks is generally referred to as Side Channel Attacks (SCA) and has received a huge interest since then. Some techniques are based on analyzing the power consumption of the cryptographic device, like Simple Power Analysis (SPA) or Differential Power Analysis (DPA) [7]. Others are based on analyzing errors during the execution of a cryptographic computation on the device, like Differential Fault Analysis (DFA) [3, 4]. These techniques may be applied without distinction to public or secret key cryptosystems. Recently a large variety of attacks and countermeasures has been proposed. However the field is now fairly well understood and naive attacks are unlikely to work against devices implementing recent countermeasures.

Therefore new directions for more sophisticated attacks are being investigated, like Higher-Order DPA for instance [9]. Many new attacks combine "traditional" cryptanalysis techniques (coming from block cipher or hash function cryptanalysis for instance) with the use of side channel information. A good example was given in 2003 by Schramm, Wollinger and Paar [12]. They proposed a

Collision Attack (CA) against DES [10] based on techniques from classical "collision attacks" against hash functions. Their attack is based on the observation that an internal collision on 3 adjacent S-boxes during a DES computation can be caused with a reasonable probability. They also gave experimental evidences that such collisions could be detected using the power consumption curves of a microcontroller. It is also interesting to notice that this technique has a close link with differential attacks against DES. Independently another CA was proposed by Wiemers [13]. It is more efficient than Schramm *et.al.*'s attack and is dedicated against DES as well. Unfortunately it has not been published so far.

The difference between DPA and CA lies in the underlying assumptions and mostly on the time scale of the analysis. Both attacks consider the correlation between some intermediate data and the corresponding power consumption curve. However, compared to usual DPA, CA focuses on larger variables (typically the input of the Feistel round function) at a larger time scale (a long sequence of instructions is analyzed). Initially CA have been applied against DES but applications have been reported recently against AES [11] and even in the field of public key cryptosystems [5]. These attacks present a particular interest because they are likely to resist against countermeasures devised specifically against DPA. Since they consider a larger time scale, countermeasures operating only at a local level might not be sufficient.

In this paper, we propose a more generic and more efficient CA. Rather than limiting our analysis to collisions, we also take into account "almost collisions", *i.e.* internal states which are extremely similar. Such events result in almost identical sequences of instructions. We choose sparse input differences that either vanish or remain sparse during several rounds. Thus we use techniques coming from differential cryptanalysis against block ciphers [2]. We show that Feistel ciphers are particularly weak regarding these new attacks.

In the Section 2, we describe a basic and generic collision attack on the second round of Feistel ciphers (with application to DES). Then, we propose an improved attack using "almost collisions" occurring in the following rounds of encryption. Finally, we present experimental results obtained with DES implemented in software on a smart-card.

2 Collision Attacks Against Feistel Ciphers

Two CA against DES have been proposed recently. In [12], it is described how to obtain and detect collisions on 3 adjacent S-boxes in the first round of DES. It is also suggested that the same method could be applied to other Feistel ciphers. Actually this attack is nice but not optimal. In [13], another CA dedicated against DES, more efficient, is briefly presented. In this section we describe a generic framework for CA against Feistel ciphers. Our description is an improvement and a generalization of these previous works.

A Feistel cipher is an iterated block cipher of size $2n$ bits where the internal state is split in two halves (L, R). The round function F operates on n bits and the next state (L', R') is computed by :

$$L' = R'$$
$$R' = L \oplus F(R)$$

For most Feistel ciphers, the round function F has 3 layers

- the addition of a subkey K.
- a non-linear layer denoted NL (*e.g.* built with several S-boxes)
- a linear application denoted $\mathcal{L}$

CAMELLIA [1] and DES [10] are examples of such a construction (we can omit the expansion in DES for the moment).

The model We assume that an attacker has access to the power consumption of a cryptographic device where some Feistel cipher is implemented without specific countermeasures. In addition, we suppose that this attacker chooses the plaintext introduced.

Although he is not able to tell from power consumption curves the values manipulated during the computation, the attacker is generally able to tell when a collision occurs. Indeed a collision usually results in two identical sequences of instructions. Hence the power consumptions curves are likely to be very similar. This assumption is reasonable as long as the corresponding computation takes many clock cycles and depends greatly on the value of the operand. For instance, we assume that a collision on the inputs of the round function F can be detected. This assumption has already been verified experimentally in [11–13]. In Section 4, we describe our own experimental results against DES implemented on a smart-card. These results comfort the validness of the previous assumption.

The attack The general idea can be stated as follows : introduce chosen differences in each branch of the Feistel that will vanish in the input of the second round function. Obviously these methods use many original ideas from differential cryptanalysis [2]. For instance, a classical result, in the case of DES, is the existence of differences on 3 adjacent S-boxes which give the same output. This idea was exploited by Schramm *et. al.* in [12].

We call δ_R the difference introduced in the right branch of the Feistel (respectively δ_L in the left branch) and Δ the output difference of the first round function. The goal in this attack is to cancel out differences on the input R_1 of the second round function. Thus we want $\Delta = \delta_L$. If this happens, we hope to detect collisions by looking at the power consumption during the second round. This scenario is summarized in Figure 1

The attack described in [12] is based on the extreme case $\Delta = \delta_L = 0$. This approach is successful in the case of DES. However, most recent Feistel ciphers use bijective round functions (although it is not a requirement of the Feistel structure) so differential trails of the form

$$\delta_R \xrightarrow{F} \Delta = 0$$

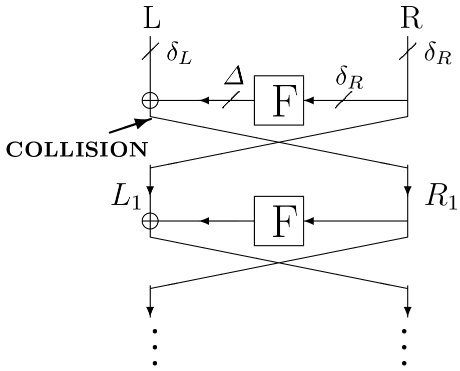

Fig. 1. Scenario of the basic collision attack

do not exist. Actually even in the case of DES this approach is not extremely efficient since about 140 messages are needed in average to obtain one collision. A more efficient approach (also used in [13]) is to introduce a low-weight difference δ_R such that only one S-box is active[1] and to cancel out this difference using δ_L. This method applies to a generic Feistel cipher, as represented in Figure 2 (where dashed areas represent differences).

We call δ_{int} the intermediate difference between layers $\mathcal{L}$ and NL. This difference is clearly limited to one S-box. Thus δ_{int} takes only 2^r different values where r is the output dimension of the S-box. We call $\delta_{int}(1), \ldots, \delta_{int}(2^r)$ these values. Looking at the coordinate on each S-box, we can write equivalently, for all i

$$\delta_{int}(i) = (i, 0, \ldots, 0)$$

Although Δ it is not necessarily limited to one S-box, it can take only 2^r values since

$$\Delta = \mathcal{L}(\delta_{int})$$

Now, the attacker tries to eliminate Δ by playing with δ_L. To that purpose, he picks a sparse δ_R which activates only one S-box and introduces the corresponding plaintexts in the block cipher :

- $P_i = (L \oplus \mathcal{L}(i, 0, \ldots, 0), R)$ for $i = 1 \ldots 2^r$
- $P'_i = (L \oplus \mathcal{L}(i, 0, \ldots, 0), R \oplus \delta_R)$ for $i = 1 \ldots 2^r$

This sums up to 2^{r+1} chosen plaintexts. Between two plaintexts P_i and P'_j, the difference in the output of the first round function is of the form

$$\Delta = \mathcal{L}(x, 0 \ldots, 0)$$

[1] In the context of differential cryptanalysis, "active" generally means that at least one input bit differs

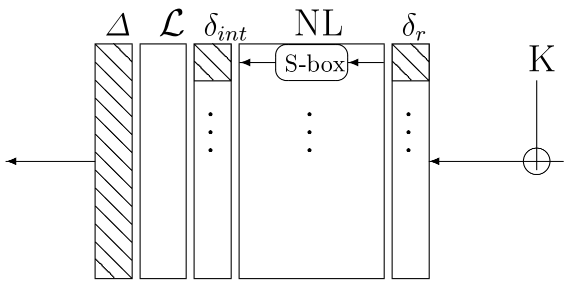

Fig. 2. The differential trail

for some value x depending only on K, R and δ_R (and not on i, j). Besides, if $i \oplus j = x$, there is a collision on R_1 because differences coming from the left branch and right branch cancel out

$$\begin{aligned}\delta_L &= \mathcal{L}(i, 0, \ldots, 0) \oplus \mathcal{L}(j, 0, \ldots, 0) \\ &= \mathcal{L}(i \oplus j, 0, \ldots, 0) \\ \Delta &= \mathcal{L}(x, 0, \ldots, 0)\end{aligned}$$

Analysis We built a set of 2^{r+1} plaintexts among which 2^r pairs $(P_i, P'_{i \oplus x})$ yield a collision on the input of the second round function. This method is much more efficient than the attack described in [12] (see the summary Table 1). In fact it is almost optimal since all available plaintexts can be useful to detect collisions.

Attack	Specificity	Active S-boxes	Block ciphers	Plaintexts/Coll.
Schramm *et. al.* [12]	$\delta_L = 0$	3	DES	140
Wiemers [13]	-	1	DES	32
this paper (basic)	-	1	any Feistel	2^r
this paper (improved)	-	1	any Feistel	$2^{1+r/2}$
this paper	-	1	DES	8

Table 1. Summary of collision attacks

The result of observing **any** of these 2^r collisions is to leak x (which gives a simple condition on a few bits from the subkey K). Since one collision is sufficient, a simple improvement is to reduce the number of plaintexts. Indeed the attacker can encrypt only the $2^{\frac{r}{2}}$ plaintexts P_i such that

$$i = \underbrace{0 \cdots 0}_{\frac{r}{2} \text{ bits}} * \cdots *$$

and the $2^{\frac{r}{2}}$ plaintexts P_j such that

$$j = * \cdots * \underbrace{0 \cdots 0}_{\frac{r}{2} \text{ bits}}$$

Here the XOR difference $i \oplus j$ spans the 2^r possible values, which guarantees that the value x we are looking for is reached once. Thus we can build a set reduced to

$$2^{\frac{r}{2}} + 2^{\frac{r}{2}} = 2^{1+\frac{r}{2}}$$

messages that yields exactly one collision. If this collision is detected, the attack succeeds. How to recover the full secret key depends highly on the key schedule, but this attack can be iterated on all S-boxes, then on the following rounds once the first subkey is entirely leaked. Furthermore, since r is typically small (from 4 to 8 bits), the number of required messages is usually reasonably small.

The case of DES Applying this generic attack to DES is straightforward. The only difference between DES and a "generic" cipher is the expansion function which has no effect on the attack. As a direct application we can build a set of $2^r = 32$ messages (since $r = 4$ bits is the output size of the DES S-boxes). Among these messages we expect 16 collisions in the second round function. As we mentioned previously, only $2 \cdot \sqrt{16} = 8$ messages are sufficient in order to guarantee the existence of a single collision.

Each collision provides a simple condition on key bits (it is a differential condition on a S-box, equivalent to the knowledge of 4 key bits). So, roughly 14 collisions are needed to expose the full key. This corresponds naively to $14 \times 8 = 112$ messages. In case less messages are available, a trade-off with exhaustive search is also possible. This result is among the most efficient side channel attacks against DES.

Similar results could be obtained for other Feistel ciphers, including CAMELLIA [1] and MISTY1 [8], both selected by the European NESSIE project.

3 An Improvement Based on "Almost-Collisions"

The previous attack exploits only power consumption curves corresponding to the second round by detecting internal collisions. In our experiments with DES, we observed that curves corresponding to the following rounds are also full of information. Indeed internal states are often very similar because of the particular form of the plaintexts. Such events - that we call "almost collisions" - are almost as easy to detect as actual collisions. In this section, we describe improved attacks based on "almost collisions".

3.1 Motivation

In the model of Section 2, we supposed that internal collisions could be detected directly from power consumption curves. Hence we gave corresponding estimates for the number of messages required. However in a practical setting, it often turns out that observations are not as good as expected. For instance, countermeasures may focus on the first rounds which are known to be critical in many attacks. Sometimes the measurements obtained are also noisy for practical reasons. Hence it often turns out that observations contain a larger amount of background noise than expected. The number of messages required for an attack is accordingly increased since the noise is generally eliminated by averaging more curves.

Another possible source of problem is that collisions are not always as easy to detect as expected. Indeed even when a collision does not occur at the end of round 1, the inputs of round 2 might still be almost identical if the diffusion of the cipher is slow. We call such a situation an **almost collision**. This notion can just be seen as a shortcut for "differences with a low hamming weight and few active S-boxes".

From a practical point of view, it is well-known that electric consumption is often correlated with the hamming weight or the hamming distance (*i.e.* the numbers of bits flipped between the previous state and the actual state). This property is often used for Simple Power Analysis or Differential Power Analysis [7]. Therefore, almost collisions are likely to result in similar power consumption curves since they correspond to differences with low hamming weight. Practical results of Section 4 illustrate that this assumption is correct. The consequent problem is that distinguishing a collision from an almost collision at round 2 is not an easy to task. To improve this analysis, we wish to take into account all available information. In particular, power consumption of the third and fourth round should be considered. Since plaintexts introduced are extremely similar, these rounds do not correspond to just random computations. Indeed, internal states can remain abnormally similar during several rounds (*i.e.* they differ only on a small number of bits). So almost collisions may be helpful if we consider the rounds number 3 or 4 of encryption. In fact, the number of active bits and S-boxes at these rounds furnish good indicators of the presence of a collision at round 2. Actually they turn out to be even more reliable than the round 2 curves themselves. In the next sections we analyze these indicators.

3.2 Differential properties of rounds 3 and 4

Basically the attacker compares two encryptions corresponding to plaintexts P_i and P'_j using notations of Section 2. His goal is to distinguish efficiently between two situations

- a collision at round 2 (*i.e.* $i \oplus j = x$)
- no collision at round 2 (*i.e.* $i \oplus j \neq x$)

For round number t, we call Δ_t the difference on the inputs of the round function F. Similarly, L_t and R_t denote the left and right branch of the Feistel structure at the end of round t for the plaintext P_i (that we write (L_0, R_0) by convention). Like in Section 2, the input difference is written (δ_L, δ_R). In case of a collision, differences on the first rounds of encryption can be expressed as follows :

Round t	Encryption of P_i	Encryption of P'_j	Difference Δ_t
1	(L_0, R_0)	$(L_0 \oplus \delta_L\ ,\ R_0 \oplus \delta_R\ )$	δ_R
2	(L_1, R_1)	$(L_1 \oplus \delta_R\ ,\ R_1\ )$	0
3	(L_2, R_2)	$(L_2\ ,\ R_2 \oplus \delta_R\ )$	δ_R
4	(L_3, R_3)	$(L_3 \oplus \delta_R\ ,\ R_3 \oplus \Delta_4\ )$	Δ_4

Table 2. Difference propagation after a collision

Thus, differences on round 2, 3 and 4 can be expressed as

$$\begin{aligned}\Delta_2 &= 0\\ \Delta_3 &= \delta_R\\ \Delta_4 &= F(R_2) \oplus F(R_2 \oplus \delta_R)\end{aligned}$$

Since δ_R has only one active S-box, both Δ_3 and Δ_4 correspond to "almost collisions" where the hamming weight is low and few S-boxes are active. In opposition, when no collision occurs, differences are more complex :

Round	Encryption of P_i	Encryption of P'_j	Difference Δ_t
1	(L_0, R_0)	$(L_0 \oplus \delta_L\ ,\ R_0 \oplus \delta_R\ )$	δ_R
2	(L_1, R_1)	$(L_1 \oplus \delta_R\ ,\ R_1 \oplus \Delta_2\ )$	Δ_2
3	(L_2, R_2)	$(L_2 \oplus \Delta_2\ ,\ R_2 \oplus \Delta_3\ )$	Δ_3
4	(L_3, R_3)	$(L_3 \oplus \Delta_3\ ,\ R_3 \oplus \Delta_4\ )$	Δ_4

Table 3. Difference propagation without collision

Differences on round 2, 3 and 4 can be expressed as

$$\begin{aligned}\Delta_2 &= F(R_0) \oplus F(R_0 \oplus \delta_R)\\ \Delta_3 &= F(R_1) \oplus F(R_1 \oplus \Delta_2)\\ \Delta_4 &= F(R_2) \oplus F(R_2 \oplus \Delta_3)\end{aligned}$$

Here, Δ_2 is quite sparse since δ_R has only one active S-box. However, the hamming weight of Δ_3 and Δ_4 can be much higher due to the diffusion properties of the block cipher. In the next section, we give estimates of these indicators in the case of DES.

3.3 Estimating the indicators for DES

Our focus now is to evaluate the hamming weight and the number of active S-boxes of Δ_2, Δ_3 and Δ_4, in two distinct cases (depending on an eventual collision at round 2). These indicators depend on the diffusion properties of DES and the differential properties of its S-boxes.

We call N_i the number of active bits in Δ_i and n_i the corresponding number of active S-boxes. First we give expected values using simple heuristic arguments. Then we give average values obtained experimentally.

Theoretical estimates First, we suppose that a collision occurs at round 2. Thus we know that $\Delta_2 = 0$ and $\Delta_3 = \delta_R$ (which has only one active S-box). Hence

$$\begin{array}{ll} N_2 = 0 & n_2 = 0 \\ N_3 = 1 \text{ or } 2 & n_3 = 1 \end{array}$$

Since Δ_4 is the image of input difference δ_R by the round function, its hamming weight is in the range from 1 to 4 with average value $N_4 = 2.5$. Besides each bit in DES internal state is involved in 1.5 S-boxes in average, so we expect

$$n_4 = 2.5 \times 1.5 = 3.75$$

When no collision is observed at round 2, a similar analysis can be conducted. The differential trail is of the form

$$\delta_R \xrightarrow{F} \Delta_2 \xrightarrow{F} \Delta_3 \xrightarrow{F} \Delta_4$$

Thus the expected values are

$$\begin{aligned} N_2 &= 2.5 \\ n_2 &= 2.5 \times 1.5 = 3.75 \\ N_3 &= 3.75 \times 2.5 = 9.375 \end{aligned}$$

At this point, all S-boxes are likely to be active in the inputs of round 3. So we expect n_3 and n_4 close to 8 and N_4 close to 16.

Practical estimates We obtained practical results for DES by performing a statistical simulation on a PC. Our basic experiment is to pick a random δ_R which only one active S-box, and a random plaintext P. We compute the first 4 rounds of encryption of P and $P \oplus (0, \delta_R)$ and observe the average values of indicators. After 10 millions experiments, we obtained the results described in Table 4.

Actually these results are even slightly better than the expected values. In rounds 3 and 4 we clearly observe an important difference between the two cases “collision at round 2” and “no collision at round 2”.

3.4 Analysis

From Table 4 we observe that the difference on the indicators is actually much more significant in round 4 than in round 2. For instance, looking at the number of active bits in round 2, the difference we try to detect is between 0 bits (when

Round	Collision	No Collision
2	$N_2 = 0$ $n_2 = 0$	$N_2 = 2.349$ $n_2 = 3.534$
3	$N_3 = 1.333$ $n_3 = 1$	$N_3 = 9.009$ $n_3 = 6.968$
4	$N_4 = 2.358$ $n_4 = 3.551$	$N_4 = 15.150$ $n_4 = 7.817$

Table 4. Average value of the indicators for DES

a collision occurs) and an average 2.349 bits (in the other case). The difference is quite small, so power consumption curves are likely to remain quite similar in both cases. However, looking at round 4, there are about 2.358 active bits in one case against 15.150 in the other. This difference is much more significant and thus easier to detect.

Our analysis is comforted by the results obtained in Section 4. In the case of DES, rounds 3 and 4 are better indicators of a collision than the round 2 itself. This is due to the slow diffusion of DES : when no collision happens at round 2 ($i \oplus j \neq x$), the difference remains quite sparse mostly because the linear layer is just a permutation of bits.

If this permutation was replaced by a linear application with better diffusion (the Mix-Column function of AES for instance) or if we considered a Feistel cipher with good diffusion (like CAMELLIA), the analysis would be different. Collisions would be easier to distinguish using the round 2 or 3, but more difficult using round 4 because of the full diffusion reached in both cases. This is summarized in Table 5.

Round	Slow diffusion (DES)	Good diffusion (CAMELLIA)
2	difficult	easy
3	easy	easy
4	easy	difficult

Table 5. Efficiency of collision detection

To conclude, we described a thiner analysis of collision attacks using differential properties, mostly by taking into account “almost collisions”. We showed that better indicators can be found to detect collisions. These improvements are extremely helpful when realizing a concrete side channel attack as we demonstrate in Section 4. We think such methods may also be helpful to defeat countermeasures which focus on protecting the second round of encryption.

4 Experimental Results

In order to verify the previous analysis we implemented a CA against DES implemented in software on a smart-card. This smart-card used classical hardware countermeasures :

- variable internal clock
- electric noise (random peaks of power)

We managed to detect collisions despite these countermeasures. The trickiest part was to get rid of the "random" peaks of power. Fortunately these peaks were not truly random (they were strongly correlated with the external clock) and were eliminated by analyzing several samples for the same encryption (*i.e.* 5 samples, but even 2 samples could be sufficient in practice). We took into account only the smallest power consumption among these samples, in order to eliminate the peaks of "over-consumption". After this preliminary work, we applied our analysis to **the full power trace of each round** (the rounds are very easy to distinguish). More precisely, we were able to identify which portions are really meaningful inside each round (namely where are located the S-box computations, etc . . .) but did not exploit it. Indeed we want to point out that collisions can be detected very simply and very efficiently.

4.1 The Attack Setting

In order to actually mount the attack, we need to introduce an appropriate set of plaintexts and detect at least one collision at round 2. As described in Section 2, 8 messages are sufficient to guarantee a collision. However we used here the full set of 32 messages described in Section 2. This simplifies the attack since we can process more data. Concretely our attack algorithm is the following

- Guess the value of x.
- For each x, identify the 16 pairs of plaintexts that should give a collision.
- For each pair of plaintexts, compute the difference Δ_{power} of power consumption curves [2].
- Average these 16 differences.

The correct value of x should yield the smallest average difference. The result obtained for round 2 are summarized in Table 6. The unit of this average value has little significance. Hence we just picked as a reference the minimal value and expressed the others as a ratio regarding this minimum.

Actually large portions of the curves are useless for this analysis (for various reasons their power consumption depends little on the arguments) and behave just like noise in practice.

[2] Our curves contain of course only a finite number of points corresponding to the electric consumption at instants t_i. The difference of consumption between two curves C and C' is by convention

$$\Delta_{\text{power}} = \sum_i (C(t_i) - C'(t_i))^2$$

Value of x	Average difference	Value of x	Average difference
0	134.26%	8	132.11%
1	121.50%	9	109.11%
2	121.86%	10	118.59%
3	113.57%	11	100%
4	140.38%	12	130.60%
5	131.55%	13	114.81%
6	131.73%	14	125.39%
7	120.70%	15	110.79%

Table 6. Average differences (correct value is $x = 11$)

4.2 Using Almost Collisions

In this section we implement the attack based on almost collision. Thus we analyze power consumption curves at rounds 3 and 4. After a collision at round 2, these curves remain quite similar, as predicted. This yields excellent results in Table 7, even better than those obtained with round 2. It comforts the assumption that almost collisions can be used as an efficient indicator.

Value of x	Average diff. for round 3	Average diff. for round 4	Value of x	Average diff. for round 3	Average diff. for round 4
0	156.26%	146.01%	8	153.38%	154.61%
1	143.45%	146.86%	9	132.05%	143.50%
2	134.32%	136.17%	10	126.01%	131.80%
3	125.03%	136.99%	11	100%	100%
4	160.36%	148.64%	12	150.70%	143.59%
5	149.98%	136.95%	13	139.99%	146.65%
6	144.10%	143.79%	14	134.46%	129.78%
7	133.34%	140.02%	15	121.11%	131.44%

Table 7. Average differences (correct value is $x = 11$)

To illustrate this attack, we represented a significant portion of round 4 for 3 plaintexts, among which 2 correspond to an almost collision (see Figure 3). The 2 corresponding curves are in average closer to each other than the third one. However some portions (like the right half of Figure 3) are more significant than the others (the left part of Figure 3 is very noisy).

At a larger scale, it is funny to notice that the useful portions of curves are positioned differently depending on the significant indicator. For instance the best indicator at round 3 is the number of active S-boxes (see Table 4) while, at round 4, the best indicator is the number of active bits. Our experiments have shown that these indicators reflect to different portions of each round (roughly, the beginning of round for active bits and the end of round for active S-boxes).

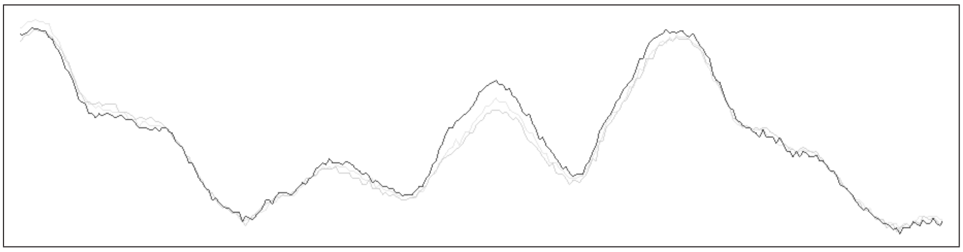

Fig. 3. Three curves corresponding to round 4

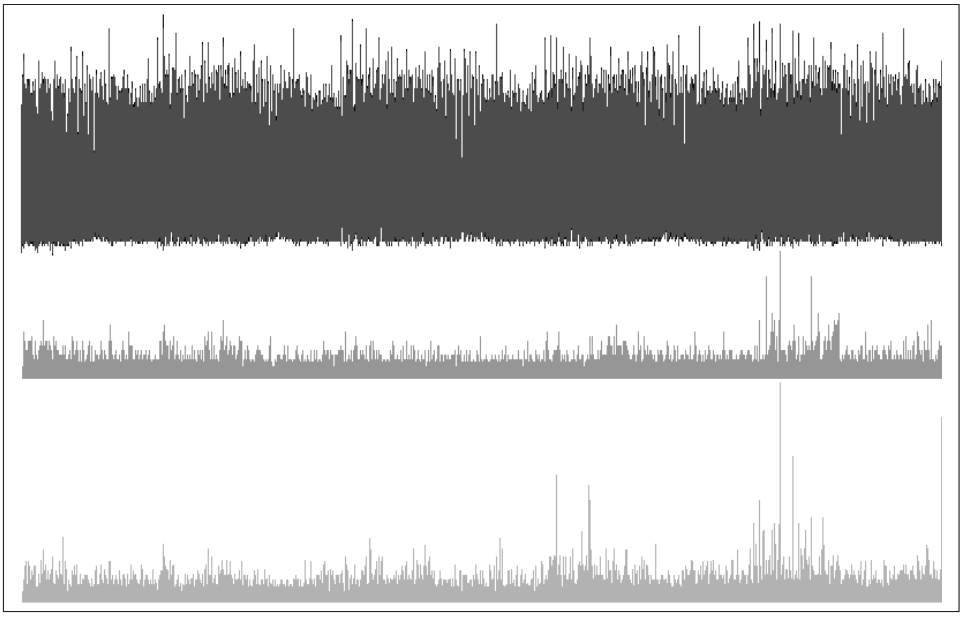

Fig. 4. The whole power consumption curves (round 1 to 4) and the corresponding differences

We have represented in Figure 4, the whole computation for the same plaintexts than those of Figure 3. In addition to the power consumption curves (represented on top but they are not very speaking), we represented a “wrong” difference (in the middle) and the “good” difference (at the bottom). This “good” difference corresponds to the almost collision. One observes that the average value of theses two additional curves increases along the computation. This is simply due to the diffusion of the input difference. Besides, the “good” difference curve has larger peaks than the “wrong” one, especially for rounds 3 and 4. Hence these rounds prove to be better indicators of a collision than the round 2 itself.

4.3 Summary

We have demonstrated that a thin analysis of the smart-card behavior at rounds 3 and 4 can lead to improved attacks, even when really few messages are available or a large amount of background noise. The remarkable thing with such attacks is that the curves for each round have been handled as a whole. Nevertheless an important bias (resulting from a collision at round 2) can be observed experimentally.

Therefore countermeasures limited to a local protection are unlikely to work against such “large-scale” attacks. Besides protecting only the first or second round with ad-hoc countermeasures is not sufficient. CA may exploit information up to round 4 or 5 depending on the diffusion speed. Countermeasures should modify the execution deeply. For instance, methods based on splitting or masking are the most likely to protect against CA. However their resistance against advanced versions of CA should be further investigated.

5 Conclusion

We described new methods for enhancing collision attacks. First we proposed a generic collision attack against Feistel ciphers which requires fewer messages than previous results and can be applied in many cases. Secondly, we suggested to improve collision attacks by considering several rounds of encryption instead of restricting the analysis to the first two rounds (as it is done by most side channel attacks). Indeed we showed that almost collisions - *i.e.* abnormally similar internal states - may appear in the collision attack scenario. They furnish better indicators than those used by previous attacks. Our experiments against DES implemented on a smart-card confirm our theoretical analysis.

Acknowledgments

We would like to thank Andreas Wiemers for some helpful discussions. We also thank Rémy Daudigny and the members of the LSC laboratory for helping us in the experimental work and capturing the power traces.

References

1. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita. Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In E. Tavares, editor, *Selected Areas in Cryptography – 2000*, volume 2012 of *Lectures Notes in Computer Science*, pages 39–56. Springer, 2001.
2. E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In A. Menezes and S. Vanstone, editors, *Advances in Cryptology – Crypto'90*, volume 537 of *Lectures Notes in Computer Science*, pages 2–21. Springer, 1990.
3. E. Biham and A. Shamir. Differential Fault Analysis of Secret Key Cryptosystems. In B. Kaliski, editor, *Advances in Cryptology – Crypto'97*, volume 1294 of *Lectures Notes in Computer Science*, pages 513–525. Springer, 1997.
4. D. Boneh, R. DeMillo, and R. Lipton. On the Importance of Checking Cryptographic Protocols for Faults (Extended Abstract). In W. Fumy, editor, *Advances in Cryptology – Eurocrypt'97*, volume 1233 of *Lectures Notes in Computer Science*, pages 37–51. Springer, 1997.
5. P-A. Fouque and F. Valette. The Doubling Attack – Why Upwards is Better than Downwards. In C. Walter, Ç. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems (CHES) – 2003*, volume 2779 of *Lectures Notes in Computer Science*, pages 269–280. Springer, 2003.
6. P. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Others Systems. In N. Koblitz, editor, *Advances in Cryptology – Crypto'96*, volume 1109 of *Lectures Notes in Computer Science*, pages 104–113. Springer, 1996.
7. P. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In M. Wiener, editor, *Advances in Cryptology – Crypto'99*, volume 1666 of *Lectures Notes in Computer Science*, pages 388–397. Springer, 1999.
8. M. Matsui. New Block Encryption Algorithm MISTY. In E. Biham, editor, *Fast Software Encryption – 1997*, volume 1267 of *Lectures Notes in Computer Science*, pages 54–68. Springer, 1997.
9. T. Messerges. Using Second-Order Power Analysis to Attack DPA Resistant software. In Ç. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems (CHES) – 2000*, volume 1965 of *Lectures Notes in Computer Science*, pages 238–251. Springer, 2000.
10. NIST FIPS PUB 46-3. *Data Encryption Standard*, 1977.
11. K. Schramm, G. Leander, P. Felke, and C. Paar. A Collision-Attack on AES Combining Side Channel And Differential-Attack. 2003. Submitted for Publication.
12. K. Schramm, T. Wollinger, and C. Paar. A New Class of Collision Attacks and its Application to DES. In T. Johansson, editor, *Fast Software Encryption – 2003*, volume 2887 of *Lectures Notes in Computer Science*. Springer, 2003.
13. A. Wiemers. Partial collision search by side channel analysis, 2003. Presentation at the Workshop : Smartcards and Side Channel Attacks.