On Boolean and Arithmetic Masking against Differential Power Analysis

Jean-Sébastien Coron[1] and Louis Goubin[2]

[1] Gemplus Card International, 34 rue Guynemer
Issy-les-Moulineaux, F-92447, France
jean-sebastien.coron@gemplus.com
[2] Bull SmartCards and Terminals
68 route de Versailles - BP45
78431 Louveciennes Cedex - France
Louis.Goubin@bull.net

Abstract. Since the announcement of the Differential Power Analysis (DPA) by Paul Kocher and al., several countermeasures were proposed in order to protect software implementations of cryptographic algorithms. In an attempt to reduce the resulting memory and execution time overhead, Thomas Messerges recently proposed a general method that "masks" all the intermediate data.
This masking strategy is possible if all the fundamental operations used in a given algorithm can be rewritten with masked input data, giving masked output data. This is easily seen to be the case in classical algorithms such as DES or RSA.
However, for algorithms that combine Boolean and arithmetic functions, such as IDEA or several of the AES candidates, two different kinds of masking have to be used. There is thus a need for a method to convert back and forth between Boolean masking and arithmetic masking.
In the present paper, we show that the 'BooleanToArithmetic' algorithm proposed by T. Messerges is not sufficient to prevent Differential Power Analysis. In a similar way, the 'ArithmeticToBoolean' algorithm is not secure either.
Keywords: Physical attacks, Differential Power Analysis, Electric consumption, AES, IDEA, Smartcards, Masking Techniques.

1 Introduction

Paul Kocher and al. introduced in 1998 ([10]) and published in 1999 ([11]) the concept of *Different Power Analysis* attack, also known as DPA. It belongs to a general family of attacks that look for information about the secret key of a cryptographic algorithm, by studying the electric consumption of the electronic device during the execution of the computation.

The initial focus was on symmetrical cryptosystems such as DES (see [10,14]) and the AES candidates (see [1,3,6]), but public-key cryptosystems have since been shown to be also vulnerable to the DPA attacks (see [15,5,9]).

Therefore, the research for countermeasures has considerably increased. In [6], Daemen and Rijmen proposed several countermeasures, including the insertion of dummy code, power consumption randomization and balancing of data.

Ç.K. Koç and C. Paar (Eds.): CHES 2000, LNCS 1965, pp. 231–237, 2000.

But these methods were proven to be insufficient: in [4], Chari and al. suggested that signal processing can be used by clever attackers to remove dummy code or to cancel the effects of randomization and data balancing. They propose a better approach, consisting in splitting all the intermediate variables. A similar "duplication" method was proposed as a particular case by Goubin and al. in [9]

However, these general methods generally increase dramatically the amount of memory needed, or the computation time, as was pointed by Chari and al. in [3]. Moreover, it has been shown in [8] that even inner rounds can be aimed by "Power-Analysis"-type attacks, so that the splitting should be performed on all rounds of the algorithm. This makes the issue of the memory and time computation overhead even more crucial, especially for embedded systems such as smart cards.

In [13], Thomas Messerges investigated on DPA attacks applied on the AES candidates. He developped a general countermeasure, consisting in masking all the inputs and outputs of each elementary operations used by the microprocessor. This generic technique allowed him to evaluate the impact of these countermeasures on the five AES algorithms.

This masking strategy is possible if all the fundamental operations used in a given algorithm can be rewritten with masked input data, giving masked output data. This is easily seen to be the case for the DES algorithm, because a single masking (using the XOR operation) can be used throughout the computation of the 16 rounds. For RSA, a masking using the multiplication operation in the multiplicative group modulo n is also sufficient.

However, for algorithms that combine Boolean and arithmetic functions, two different kinds of masking have to be used. There is thus a need for a method to convert back and forth between Boolean masking and arithmetic masking. This is typically the case for IDEA [12] and for three AES candidates: MARS [2], RC6 [16] and TWOFISH [17].

Thomas Messerges proposed in [13] an algorithm in order to perform this conversion between a "$\oplus$ mask" and a "+ mask". Unfortunately, we show in the present paper that the 'BooleanToArithmetic' algorithm proposed by T. Messerges is not sufficient to prevent Differential Power Analysis. In a similar way, the 'ArithmeticToBoolean' algorithm is not secure either. A detailed attack is described.

2 The "Differential Power Analysis" Attack

The "Differential Power Analysis" attack, developped by Paul Kocher and Cryptographic Research (see [10,11], see also [7]), starts from the fact that the attacker can get much more information (than the knowledge of the inputs and the outputs) during the execution of the computation, such as for instance the electric consumption of the microcontroller or the electromagnetic radiations of the circuit.

The "Differential Power Analysis" (DPA) is an attack that allows to obtain information about the secret key (contained in a smartcard for example), by performing a statistical analysis of the electric consumption records measured for a large number of computations with the same key.

Let us consider for instance the case of the DES algorithm (Data Encryption Standard). It executes in 16 steps, called "rounds". In each of these steps, a transformation F is performed on 32 bits. This F function uses eight non-linear transformations from 6 bits to 4 bits, each of which is coded by a table called "S-box".

The DPA attack on the DES can be performed as follows (the number 1000 used below is just an example):

Step 1: We measure the consumption on the first round, for 1000 DES computations. We denote by E_1, ..., E_{1000} the input values of those 1000 computations. We denote by C_1, ..., C_{1000} the 1000 electric consumption curves measured during the computations. We also compute the "mean curve" MC of those 1000 consumption curves.

Step 2: We focus for instance on the first output bit of the first S-box during the first round. Let b be the value of that bit. It is easy to see that b depends on only 6 bits of the secret key. The attacker makes an hypothesis on the involved 6 bits. He computes – from those 6 bits and from the E_i – the expected (theoretical) values for b. This enables to separate the 1000 inputs E_1, ..., E_{1000} into two categories: those giving $b = 0$ and those giving $b = 1$.

Step 3: We now compute the mean MC' of the curves corresponding to inputs of the first category (*i.e.* the one for which $b = 0$). If MC and MC' show an appreciable difference (in a statistical meaning, *i.e.* a difference much greater than the standard deviation of the measured noise), we consider that the chosen values for the 6 key bits were correct. If MC and MC' do not show any sensible difference, we repeat step 2 with another choice for the 6 bits.

Note: In practice, for each choice of the 6 key bits, we draw the curve representing the difference between MC and MC'. As a result, we obtain 64 curves, among which one is supposed to be very special, *i.e.* to show an appreciable difference, compared to all the others.

Step 4: We repeat steps 2 and 3 with a "target" bit b in the second S-box, then in the third S-box, ..., until the eighth S-box. As a result, we finally obtain 48 bits of the secret key.

Step 5: The remaining 8 bits can be found by exhaustive search.

Note: It is also possible to focus (in steps 2, 3 and 4) on the set of the four output bits for the considered S-boxes, instead of only one output bit. This is what we actually did for real smartcards. In that case, the inputs are separated into 16 categories: those giving 0000 as output, those giving 0001, ..., those giving 1111. In step 3, we may compute for example the mean MC' of the curves corresponding to the last category (*i.e.* the one which gives 1111 as output). As a result, the mean MC' is computed on approximately $\frac{1}{16}$ of the curves (instead of approximately half of the curves with step 3 above): this may compel us to use a number of DES computations greater than 1000, but it generally leads to a more appreciable difference between MC and MC'.

This attack does not require any knowledge about the individual electric consumption of each instruction, nor about the position in time of each of these instructions. It applies exactly the same way as soon as the attacker knows the outputs of the algorithm and the corresponding consumption curves. It only relies on the following fundamental hypothesis:

Fundamental Hypothesis*: There exists an intermediate variable, that appears during the computation of the algorithm, such that knowing a few key bits (in practice less than 32 bits) allows us to decide whether two inputs (respectively two outputs) give or not the same value for this variable.*

3 Review of Countermeasures

Several countermeasures against DPA attacks can be conceived. For instance:

1. Introducing random timing shifts, so that the computed means do not correspond any longer to the consumption of the same instruction. The crucial point consists here in performing those shifts so that they cannot be easily eliminated by a statistical treatment of the consumption curves.
2. Replacing some of the critical instructions (in particular the basic assembler instructions involving writings in the carry, readings of data from an array, etc) by assembler instructions whose "consumption signature" is difficult to analyze.
3. For a given algorithm, giving an explicit way of computing it, so that DPA is provably unefficient on the obtained implementation. The masking strategy, detailed below is an example of this third kind of method.

4 The Masking Method

In the present paper, we focus on the "masking method", initially suggested by Chari and al. in [3], and studied further in [4].

The basic principle consists in programming the algorithm so that the fundamental hypothesis above is not true any longer (*i.e.* an intermediate variable never depends on the knowledge of an easily accessible subset of the secret key). In a concrete way, using a secret sharing scheme, each intermediate that appears in the cryptographic algorithm is splitted. Therefore, an attacker has to analyze multiple point distributions, which makes his task grow exponentially in the number of elements in the splitting.

In [13], Messerges applied this fundamental idea for all the elementary operations that can occur in the AES algorithms. For algorithms that combine Boolean and arithmetic functions, such as MARS, RC6 and TWOFISH, two different kinds of masking have to be used:

Boolean masking: $x' = x \oplus r$
Arithmetic masking: $x' = (x - r) \bmod 2^k$

Here the variable x is masked with random r to give the masked value x'.

The conversion from boolean masking to arithmetic masking as described in [13] works as follows:

BooleanToArithmetic
Input: (x', r) such that $x = x' \oplus r$.
Output: (A, r) such that $x = A + r$
Randomly select: $C = 0$ or $C = -1$
$B = C \oplus r$; /* $B = r$ or $b = \bar{r}$ /*
$A = B \oplus x'$; /* $A = x$ or $A = \bar{x}$ /*
$A = A - B$; /* $A = x - r$ or $A = \bar{x} - \bar{r}$ /*
$A = A + C$; /* $A = x - r$ or $A = \bar{x} - \bar{r} - 1$ /*
$A = A \oplus C$; /* $A = x - r$ /*
Return(A, r);

The conversion from the arithmetic masking to the boolean masking can be done with a similar algorithm.

The conversion from one type of masking to another should be done in such a way that it is not vulnerable to DPA attacks. The previous algorithm takes as input the couple (x', r) such that $x = x' \oplus r$. The unmasked data is x and the masked data is x'. The algorithm works by unmasking x' using the XOR operation and then remasking it using the addition operation.

The issue is that the variable x or $\bar{x}$ is computed during the execution of the algorithm. It is stated in [13] that a DPA attack will not work against this algorithm because the attacker does not know whether x or $\bar{x}$ is processed. This is true for a DPA selecting one bit of x: since x and $\bar{x}$ are processed with equal probability, the processed bit is decorrelated from the key and the single-bit DPA does not work. This is not the case if we perform a DPA with 2 selected bits, as shown in the next section.

5 A DPA Attack against the Conversion Algorithm

The attack is based on the fact that if 2 bits of x are equal, the corresponding bits are also equal in $\bar{x}$. Consequently, we modify the DPA attack described in section 2. Instead of selecting the curves from the predicted value of a given bit of x, we consider 2 bits and divide the power samples into 2 groups: in the first group, the 2 bits are equal, and in the second group they are distinct. The classification is not affected by the processing of x and $\bar{x}$. Consequently, if the power consumption when 2 bits are equal differs from the power consumption when 2 bits are distinct, the 2-bits DPA works: the proper key hypothesis should show a peak, while the others will be mostly flat, so that the all the key bits will be recovered.

Consider the four conditional laws for the power consumption and denote their respective mean values $\mu_{00}, \mu_{01}, \mu_{10}, \mu_{11}$. For the proper key hypothesis, the mean value of the first group is:

$$\frac{\mu_{00} + \mu_{11}}{2}$$

and the mean value of the second group is:

$$\frac{\mu_{01} + \mu_{10}}{2}$$

The mean of the difference between the two groups is thus:

$$D = \frac{\mu_{00} + \mu_{11} - \mu_{01} - \mu_{10}}{2} \quad (1)$$

Consequently, the 2-bits DPA works if $D \neq 0$.

We would like to stress that our attack is not a high-order DPA. A high-order DPA [11] consists in looking at joint probability distributions of multiple points in the power signal. As shown in [4], a high-order DPA attack requires a number of experiments exponential in the number of points considered. Instead, our attack concentrates on a single point in the power signal. Consequently, the number of required experiments should be of the same order as for a single-bit DPA.

6 Conclusion

We have described a DPA attack against the conversion algorithm proposed in [13]. Our attack is a straighforward extension of the classical DPA attack. We did not have time to perform the experiments to validate our attack in practice but we think that the threat is real and such algorithm for converting from boolean masking to arithmetic masking should be avoided.

A natural research direction is to find an efficient algorithm for converting from boolean masking to arithmetic masking and conversely, in which all intermediate variables are decorrelated from the data to be masked, so that it is secure against DPA.

Acknowledgements. We would like to thank the anonymous referees for their helpful comments.

References

1. Eli Biham and Adi Shamir, "Power Analysis of the Key Scheduling of the AES Candidates", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, March 1999. `http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm`.
2. C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S.M. Matyas, L. O'Connor, M. Peyravian, D. Safford, and N. Zunic, "MARS - A Candidate Cipher for AES", NIST AES Proposal, Jun 98.
3. Suresh Chari, Charantjit S. Jutla, Josyula R. Rao and Pankaj Rohatgi, "A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, `http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm`, March 1999.
4. Suresh Chari, Charantjit S. Jutla, Josyula R. Rao and Pankaj Rohatgi, "Towards Sound Approaches to Counteract Power-Analysis Attacks", in *Proceedings of Advances in Cryptology – CRYPTO'99*, Springer-Verlag, 1999, pp. 398-412.

5. Jean-Sébastien Coron, "Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 292-302.
6. John Daemen and Vincent Rijmen, "Resistance Against Implementation Attacks: A Comparative Study of the AES Proposals", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, `http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm`, March 1999.
7. John Daemen, Michael Peters and Gilles Van Assche, "Bitslice Ciphers and Power Analysis Attacks", in *Proceedings of Fast Software Encryption Workshop 2000*, Springer-Verlag, April 2000.
8. Paul N. Fahn and Peter K. Pearson, "IPA: A New Class of Power Attacks", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 173-186.
9. Louis Goubin and Jacques Patarin, "DES and Differential Power Analysis – The Duplication Method", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 158-172.
10. Paul Kocher, Joshua Jaffe and Benjamin Jun, "Introduction to Differential Power Analysis and Related Attacks", `http://www.cryptography.com/dpa/technical`, 1998.
11. Paul Kocher, Joshua Jaffe and Benjamin Jun, "Differential Power Analysis", in *Proceedings of Advances in Cryptology – CRYPTO'99*, Springer-Verlag, 1999, pp. 388-397.
12. X. Lai and J. Massey, "A Proposal for a New Block Encryption Standard", in *Advances in Cryptology - EUROCRYPT '90 Proceedings*, Springer-Verlag, 1991, pp. 389-404.
13. Thomas S. Messerges, "Securing the AES Finalists Against Power Analysis Attacks", in *Proceedings of Fast Software Encryption Workshop 2000*, Springer-Verlag, April 2000.
14. Thomas S. Messerges, Ezzy A. Dabbish and Robert H. Sloan, "Investigations of Power Analysis Attacks on Smartcards", in *Proceedings of USENIX Workshop on Smartcard Technology*, May 1999, pp. 151-161.
15. Thomas S. Messerges, Ezzy A. Dabbish and Robert H. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 144-157.
16. R.L. Rivest, M.J.B. Robshaw, R. Sidney and Y.L. Yin, "The RC6 Block Cipher", v1.1, August 20, 1998.
17. B. Schneier, J. Kemsey, D. Whiting, D. Wagner, C. Hall and N. Ferguson, "Twofish: A 128-Bit Block Cipher", AES submission available at: `http://www.nist.gov/aes`.