The Journal of Systems and Software 84 (2011) 753–762

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Shape analysis for power signal cryptanalysis on secure components

Frédérique Robert-Inacio [a,b,*], Alain Trémeau [c], Mike Fournigault [a], Yannick Teglia [d], Pierre-Yvan Liardet [d]

[a] Institut Matériaux, Microélectronique et Nanosciences de Provence (IM2NP) UMR CNRS 6242, France
[b] ISEN Toulon, Place Pompidou, F-83000 Toulon, France
[c] Laboratoire Hubert Curien UMR CNRS 5516, 18 Rue du Professeur Benoît Lauras, F-42000 Saint-Etienne, France
[d] ST Microelectronics, 77 Avenue O. Perroy, F-13790 Rousset, France

ARTICLE INFO

Article history:
Received 17 December 2009
Received in revised form 4 October 2010
Accepted 15 December 2010
Available online 25 December 2010

Secure component
Reverse engineering
Pattern recognition
Similarity parameter
Cryptography

ABSTRACT

This paper presents an application of pattern recognition techniques in reverse engineering for smart cards. The aim of the study is to design algorithms based on shape classification and to determine instructions executed on a chip as well as processed data sets. Information is extracted from the power consumption in order to recover secret information. Then geometrical features are determined and a syntactic analysis is achieved in order to recover secret algorithms and data. Some examples are given showing how code execution can be reversed on a recent secure component. These examples are essentially focused on instruction recovery but the algorithms also work on data recovery or on a combination of both instruction and data recovery.

1. Introduction

Reverse engineering is an important field in cryptography applications (Ahn and Lee, 2006; Novak, 2003; Peeters et al., 2007). Actually it is really useful for cryptanalysis and for component design in microelectronics context. In this way it becomes possible for a designer to elaborate new counter measures in order to prevent pointed out leakages. Numerous studies have been conducted in the last 10 years about those information leakages and means to prevent them (den Hartog and de Vink, 2005). Most of them are based on signal processing methods and side-channel attacks (Kocher, 1996) are the most well-known attacks by power analysis (Kocher et al., 1999; Chari et al., 2003). Let us pay more attention to the application context. For a standard chip embedded in a smart card, different information can be measured during processing. Fig. 1 shows different attacks on a secure component. Attacks can be lead by simple power analysis (SPA) or differential power analysis (DPA) on side channel (Le et al., 2008), or can be achieved by considering electromagnetic power radiation, otherwise by detecting faults or by analysing timing behaviour (Gammel and Ruping, 2005; Wollinger et al., 2004; Brumley and Boneh, 2005).

Both attacks usually model the side-channel leakages using the so-called *Hamming weight* and *Hamming distance* models (Peeters et al., 2007), i.e. they only consider the number of bit transitions in a device as an image of its leakage. In these models, the main difference between power and electromagnetic analysis is assumed to be the fact that the latter allows space localization (Peeters et al., 2007). A lot of publications recognize the possibility to recover the signature of an instruction in a side channel trace. Except for the article of Quisquater and Samyde (2002), it seems that no article demonstrates how to automate reverse engineering of software code, using this assumption. According to Quisquater and Samyde it is possible to build a tool dedicated to parse the trace and to gather opcodes during a simple acquisition. In order to improve the decision criteria, i.e. to better recover instructions, Quisquater and Samyde proposed to use a neural network and a learning phase process. Their model is based on the assumption that each instruction gives a different trace by power analysis. In fact, this is not always the case because the *action* of the first instruction can modify the trace of the second instruction. Quisquater and Samyde (2002) showed that the signature of an instruction is an expression of its own address in memory, the data handled and sometimes the address where the data will be stored. Transferring address modifies the shape and the very last peak of an instruction power trace. In our case of study we investigate a processor which can be described with only one template for all its instructions. It is possible to carry out a recognition of the instructions using a *simple* correlation with a dictionary. Building a dictionary where each instruction is represented by a limited number of points of measurements is not a

* Corresponding author at: Institut Matériaux, Microélectronique et Nanosciences de Provence (IM2NP) UMR CNRS 6242, ISEN Toulon, Place Pompidou, 83000 Toulon, France.
E-mail address: frederique.robert@isen.fr (F. Robert-Inacio).

0164-1212/$ – see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2010.12.022

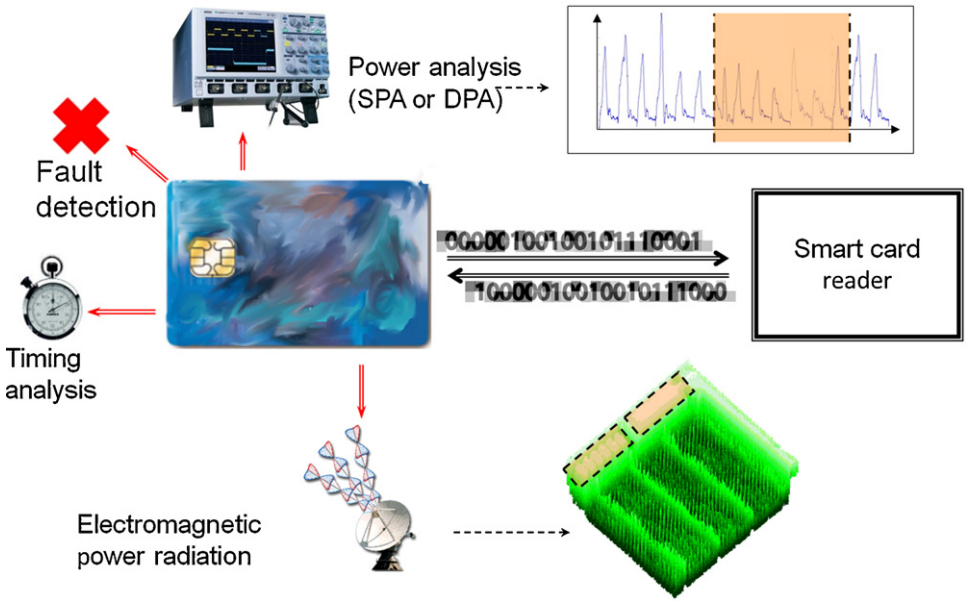

Fig. 1. Attacks on an electronic component.

difficult task. Inversely, building a correlation measure based on a shape measure is a difficult task. To face this problem Quisquater and Samyde (2002) proposed to use an automatic classifier based on a networks of neurons. Their correlation technique is based on a correlation of signatures of traces of instructions. In our approach we have considered the power signal and the shapes it generates. Rather than using a point-by-point correlation measure for the n points associated with an instruction, we propose to use a syntactic correlation measure based on the instruction shape. In other words the power signal curve bounds a shape located between the curve itself and the *X*-axis. In this way shapes are considered in their whole and not through some descriptors or distance values. The main advantage is that the discriminant power of such a measure of similarity is considerably increased as a small difference in shape (especially on the boundary) leads to a similarity value really less than 100%. In this paper our approach is demonstrated essentially for instruction recovery but it can be used for data recovery as well, as the algorithm deals with shapes whatever they represent. In order to have a full experiment using both data and instructions the database used as reference set must be enlarged to more well-known shapes including instructions processing different data sets. As shapes depend on both instructions and processed data, specific databases can be built up depending on the use purpose. Databases dedicated to data recovery and databases dealing with instruction recovery can be used either separately or simultaneously. Such databases can contain more or less elements depending on the required level of accuracy. Actually a sample step has to be defined for data sampling and then a shape per each pair of dataset and instruction needs to be computed and included in the database during the learning stage. But conditional instructions also have to be taken into account as they make the instruction retrieval more difficult (Vermoen et al., 2007). This will be the purpose of further developments. Now let us focus on instruction recovery.

Different attacks can be led in order to retrieve algorithms and data processed on a secure component (Fig. 1). These attacks can either be passive or made under stress. In our application we take into account passive attacks: mainly power analysis and in future works electromagnetic power radiation.

Fig. 2 shows some examples of such curves and shapes. Then, pattern recognition and shape classification can be achieved on the generated shapes. In order to simplify the process, each generated shape is decomposed in several elementary shapes (see Section 2). Afterwards, these elementary shapes are classified (see Section 3) and a syntactic analysis is done to interpret each cycle (see Section 4), followed by a semantic analysis determining the instruction corresponding to a set of cycles, so that executed algorithms and processed data sets could be recovered (see Section 5). Fig. 3 describes the whole process.

2. Signal decomposition

First of all, each power signal curve is cut up in a set of elementary curves so that they define elementary shapes. This

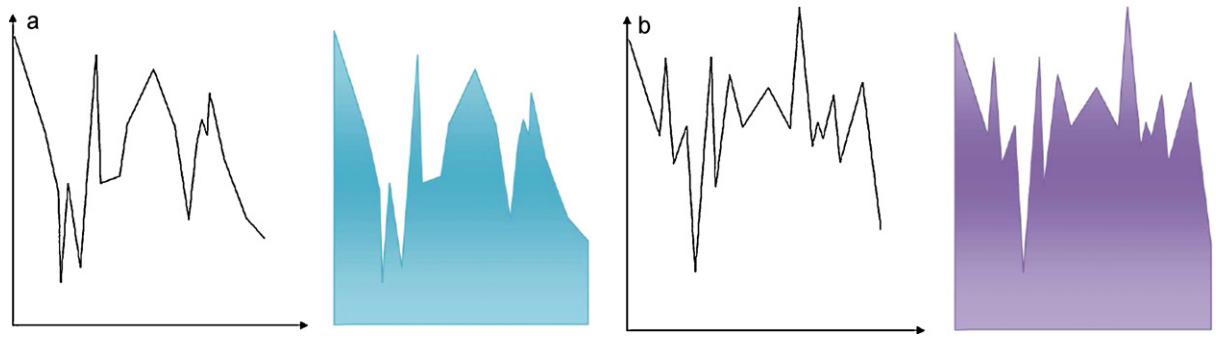

Fig. 2. Power signal curves on the left and associated shapes on the right.