USENIX Association

Proceedings of the 5th Smart Card Research and Advanced Application Conference

San Jose, California, USA
November 21–22, 2002

Automatic Code Recognition for smart cards using a Kohonen neural network

Jean-Jacques Quisquater
David Samyde
Université catholique de Louvain, UCL Crypto Group
Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium
{jjq,samyde}@dice.ucl.ac.be, http://www.dice.ucl.ac.be/crypto

Abstract

A processor can leak information by different ways. Although, the possibility of attacking smart cards by analyzing their power consumption [Kocher] or their electromagnetic radiations is now commonly accepted [Gandolfi]. A lot of publications recognize the possibility to recover the signature of an instruction in a side channel trace. It seems that no article demonstrate how to automate reverse engineering of software code, using this assumption. Our work describes a method to recognize the instructions carried out by the processor. In a general way, a classifier permits to identify the right or wrong value during the comparison of a pin code or large parts of a software code. On a few micro-controllers, using a classical correlation between the power trace and a dictionary, we show how to identify the CPU's actions. Sometimes, silicon manufacturers hide specific opcodes deliberately. The EM investigation and the template attack demonstrated by IBM, at Cryptographic Hardware and Embedded Systems 2002, rely on multivariate signal processing for electromagnetic and power traces. The method presented in this article is based on a self organizing map. On a CISC processor, it is then obvious to find a hidden instruction looking for a hole or a bad construction of the map. The case of pipelined processors is a little bit different: as they decode, execute, fetch, several parts of different opcodes at the same time, it is more difficult to recognize a specific signature.

1 Introduction

Processor's power consumption has been known for a long time by a restricted group of people in the smart card community. For security purposes, the knowledge of this side channel has been kept secret. In 1998, Paul Kocher introduced the concept of Differential Power Analysis. It was the first introduction of real signal processing for smart card attacks... Differential Power Analysis (DPA) can be explained as the correlation of two random variables. Today, people are confident with power analysis, but they also know that current measurement is not the only source of information leakage. The electromagnetic radiations can give the same result. Last year, the security group of Gemplus demonstrated that electromagnetic side channel must be taken into account seriously [Gandolfi]. The signal noise ratio of electromagnetic analysis (EMA) is much better than the signal noise ratio of power analysis. EMA measurements are very noisy. Power analysis measurements contain lower frequencies than EMA. For the same processor, the number of traces necessary to recover the secret key is reduced for EMA. The practical implementation of power analysis is very simple to realize unlike EMA. One of the big advantages of EMA is the locality principle. Using a very little sensor, it is obvious to localize exactly a specific leakage source on a processor [Quisquater]. Actual improvements of classical non intrusive analysis are linked to sophisticated signal processing [Boneh]. For differential analysis, Bayesian methods allow to recover the secret key with much less than forty traces.

On of the most important parts in differential side channel analysis, is the decision criteria. A classical countermeasure against power or electromagnetic analysis is to defeat the selection function of the attacker. In this case wrong guesses appear. An engineer very familiar with power trace of a specific processor can easily recognize each instruction carried out by the device. Thus it is possible to build a tool dedicated to parse the trace and to gather opcodes during a simple acquisition. A neural network can improve the decision criteria. With such a tool, once the learning phase done, it must be easy to recover instructions.

Two instructions executed on a Complex Instruction Set Computer processor can necessitate a different number of clock cycles. Generally a Reduced Instruction Set Computer processor execute one instruction per clock cycle. The number of clock cycles per instruction is a source of information for an attacker. Asynchronous processors do not have clock, so their actions are not easily identifiable. Anyway, it is possible to recover very specific patterns such as some memory access (charge pump) parsing power or EM traces. For a classical processor, the role of the clock is dominating, and the principal component of consumption remains is linked to the clock signal.

2 Chip depackaging

For a long time, smart cards have been famous for their tamper resistance [Anderson]. They are protected against invasive attacks. Security sensors are various, so a lot of attacks can be stopped very early. In order to discourage attackers, it is quite difficult to open the package and directly access the microprocessor without damaging it. Many traps have been introduced by designers. A lot of processors are surrounded by grids, physical parameters are checked automatically. The continuity or the resistance of these sensors indicates a correct operation mode to the processor. But the engineers do not only use passive sensors, they also introduce false radiation sources, clock jitter, low temperature and light detectors...

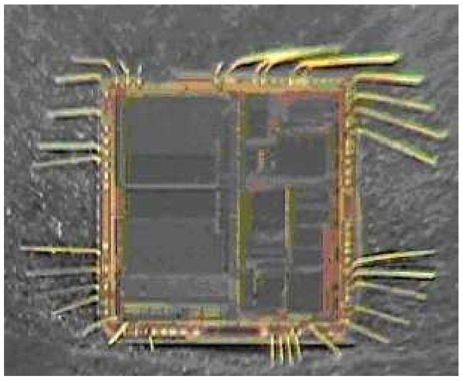

Figure 1: A depackaged processor

Recent attacks permit to cancel some sensors. With very concentrated nitric acid and organic solvent it seems to be possible to depackage a smart card or a classical processor without damaging it. When the chip is depackaged it is very easy to see regular structures or external sensors, sometimes without a specific microscope. Inside a structure, the logical gates are mixed in order to complicate the spot of a person wishing to do reverse engineering.

On figure 1, a classical processor has been opened. It is based on an eight bit architecture including a pipeline. All the instructions are done into four or eight clock cycles. On the left side, two identical banks are visible, and on the other side non regular structures are present. This pipelined processor is well known from the pay-TV pirates. It is currently extremely used with an I^2C memory for false smart cards. In general, test circuits of classical smart card are specific, it is not the case here.

3 Power measurements vs electromagnetic measurements

The electromagnetic radiations have recently been investigated as one of the new sources of side channel for the smart cards. The magnetic field is much more investigated than the electric one. The sensor is very often a coil, placed in the close field of the processor.

If we separate the spectrum into two principal components, each one dominating largely over a frequency band, the electric field from DC to 10 MHz carries information different from the magnetic field. In fact the propagated wave is not the same at all. A small capacitor or a wire simulating an antenna is enough to investigate seriously effects of the electric field. It allows to locate more precisely some parts of the chip (phase locked loop, charge pumps,). Inspecting this band also permits to recover the presence of the clock signals, often lower than 10 MHz. Some smart card integrate their own internal clocks or frequency multipliers.

The traditional power consumption analysis [Messerges] recovers the actions of the processor but does not permit to map a chip. The global consumption measurement is the sum of local consumption of each local substructures of the smart card. It may be possible with some signal processing to isolate the consumption from each component. We wish to retrieve the code executed by the processor. In order to be able to isolate the circuitry concerning instructions, we use electromagnetic radiation but particularly the electric field.

4 The pipe-line and the influence of each instruction

The processor we want to analyze contains a four-stage pipeline. Four clock cycles are necessary to the processor to carry out an instruction. But at each clock cycle it is in fact a fetch, a decode, an execution, a storage. The influence of the pipeline is then present on the side channel trace. And in an extended way, the preceding instruction modifies the consumption of the instruction in course of execution.

In order to highlight such an assertion, we have an instruction (A) executed by the processor followed by several identical instructions (Bi). This reveals that the first instruction (B1) after the instruction (A) has its

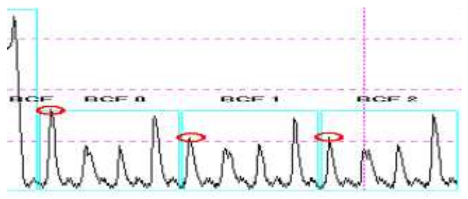

Figure 2: The interaction between two instructions

trace modified by the instruction (A). We can realize here that the first quarter of the trace is stronger if the instruction before was a Bit Clear in File register (BCF) with a pad set to one.

Instructions can have interaction with themselves. In order to validate our working hypothesis, we execute several identical instructions (Bi). Normally, all the instructions should have an identical trace. In fact it is not the case and the "action" of the first instruction modifies the trace of the second instruction. It is exactly what the figure 2 demonstrates.

The signature of each instruction contains four peaks. On figure 2, the first peak of the BCF is directly linked to the instruction just before, but not only. When BCF follows another BCF having called upon the establishment of an external pad, to one using a particular register, the first peak is different. Between the BCF 0 and the BCF 1, the difference is visible, and is only related to the external action of the preceding instruction.

5 An instruction signature

Each instruction gives a different trace by power analysis [Fahn]. But 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. The Hamming weight of each data/adress is clearly visible by electromagnetic analysis, but it is still impossible to detect 55 to a AA as their Hamming weight is the same. Using the electric field, it is possible to recover more information. Our antenna is based on a bonding wire, so if the

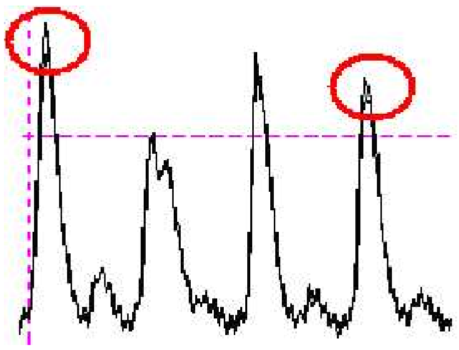

Figure 3: Two signatures with two addresses and two Hamming weight

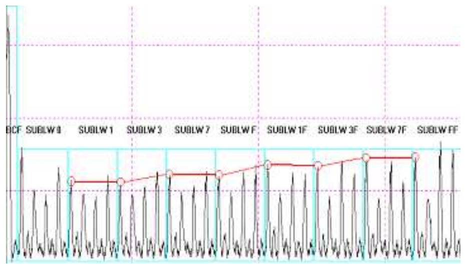

Figure 4: The influence of the address on the signature

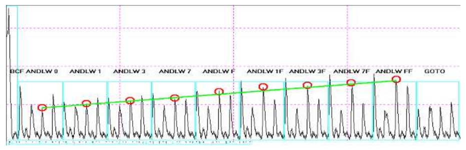

Figure 5: The influence of data's Hamming weight

architecture of the bus does not equilibrate the electric field radiated, it is possible being closer to a group of wire to identify their activity. A possible countermeasure against such an analysis is to use two wires for coding 0 and 1. It is called dual logic, two other states are free to specify alarms. (01 = 0, 10 = 1, 00 = 11 = alarm). Ross Anderson, Markus Kuhn and Sergei Skorobogatov, suggest to use this architecture to design new processors. If a pair of wire is set to 00 or 11 it products a detectable fault.

The influence of an instruction's address on its signature can be easily showed. The same couple of instructions executed several times at different addresses does not permit to obtain a constant trace. The Hamming weight involved is very different. The figure 3 shows the influence of the address on the first and last cycles of the clock. This figure represents two instructions CLear Work Register (CLRW) at two addresses with an extremely different Hamming weight. The first modification corresponds with the address execution of the current instruction, whereas the last peak is the expression of the address of the next instruction to be executed by the pipeline.

Moreover, consumption traces clearly indicate that the consecutive addresses sometimes contain an identical Hamming weight. Figure 4 details the trace of consumption of the instruction SUBstract Literal from Work register (SUBLW) for nine continuous addresses. The first clock cycle is influenced by the address of the instruction executed. Two consecutive address with the same Hamming weight generate two identical power peaks.

The Hamming weight of the manipulated data is important too [Kommerling]. On our processor, the third peak expresses this data. Figure 5 proves that quite a linear equation between the Hamming weight and the consumption exists. Of course the processor under analysis does not contain any power or electromagnetic analysis countermeasures. Advanced smart card processors are generally protected against non intrusive analysis [Coron]. In some processors the data bus is encrypted, (sometimes the address bus too). If an attacker is able to recognize patterns using the Hamming weight, he can be able to extract cryptographic keys [Kuhn]. Then to avoid such an analysis, the implementation cryptographic algorithm are well secured against power or electromagnetic analysis.

Transfering address modifies the shape and the very last peak of an instruction's power trace. The linear properties are maintained compare to the address properties. The main difference between the influence of the Hamming weight on address and data is the localization in the signature.

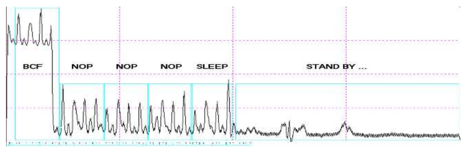

Figure 6: The sleep instruction

6 A correlation attack

The processor we decided to investigate can be described with only one template for all its instructions. But some processors do not have instructions with quite the same signature. In a few cases, the power traces [Biham] are very different for each instruction compared to the others, and their automatic recognition is common place.

In our case, for the processor studied, each instruction has a signature on a four clock cycles. On the other hand, the values of the various peaks are directly related to the opcode. It is the case of figure 6. The sleep instruction gives a characteristic signature just before the deactivation of the processor.

It is then 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 t_n points of measurement is not so difficult. Each power or EM trace is saved in the dictionary as S_n points. By multiplying each t_i point with the S_i corresponding value, and by normalizing the result, we obtain a specific opcode detector. Standardization is not necessary because if measurements are carried out in the same environment as during the creation of the dictionary, the voltages and the measured currents are the same too. The only requirement is to have a comparator triggered on a high threshold, in order to secure the false alarms.

The calculation of correlation avoids the permanent resynchronization. To isolate the known patterns from a consumption trace, it is necessary to be synchronous with the trace [Kelsey]. Our experiments for the moment are not executed in real time, and our traces are stored over for a long time by using a memory board of 1 Gigabytes. Then the values go trough a correlator which provides after a comparison/threshold the name of the instruction identified with the Hamming weight concerned. This method works with a high level recognition (better than 87%) better for CISC processor than for RISC ones. Anyway we tested it on a Z80 processor and we managed to recover more than 95% of a software. Parsing the power and EM trace, we were able to extract each pattern and to identify it, using the dictionary built just before.

Ideally we wish to be able to find the codes carried out on RISC processors, but as many instructions have very close signatures, the error rate becomes important.

7 Self organizing maps

It is then necessary to change the structure we used. We have decided to use an automatic classifier. This work can be done by a neural network.

The Kohonen's self organizing maps are based on a network of K neurons with N input. The network has K outputs. The inputs are vectors with N components, all connected completely to K neurons of the network by NK modifiable connections. The neurons of the network are placed in a two dimensional space. Each neuron has neighbors in this space. Each neuron has lateral connections according to the core of convolution of the Mexican hat operating on its neighbors.

It is supposed that the weights of modifiable connections are initially random. For an input X vector the output of the network is a y vector with K components:

$$y_i = \sum_j^{n=1} W_{ij} X_j = X^T.W_i$$

In this equation W_i is the vector weight of neuron i, i.e. the vector with N components W_{ij}. According to the vector X and the initial configuration of the weights, there is a neuron i_0 whose output is the largest. This neuron is considered as the winner.

The activity of the neurons close to the neuron i_0 is facilitated, while that of distant neurons is inhibited. After balance, the output of the network reveals a zone of dominating activity around the winning neuron, surrounded by inactive zones, or slightly active. Modifiable connections are then adjusted according to the rule:

$$\Delta W_i = \alpha y_i (X - W_i)$$

The highest values of y_i can be found in the most active zone, i.e. around the neuron i_0, the corrections are important while they are very weak in the slightly active zones and null in the inactive zones. Qualitatively, the synaptic correction tends to make the neuron i_0 more selective to the X data. Indeed, for the neuron i_0 , the output is more important, one checks before the correction:

$$\forall i \neq i_0, y_{i0} = X^T W_{i0} > y_i = X^T W_i$$

After the correction, replacing $W_i \Longleftrightarrow W_i + \triangle W_i$ we obtain:

$$\begin{aligned} y_i &= X^T.(W_i + \Delta W_i) \\ &= X^T.[W_i + \alpha X^T W_i (X - W_i)] \\ &= X^T W_i (1 + \alpha X^T (X - W_i) \end{aligned}$$

If the scalar product $X^T(X - W_i)$ is positive, after the correction of the weight, the selectivity of the neuron i_0 to the vector X is increased. The neurons linked to lateral connections, are also strongly concerned by the modification of the weights [Kohonen1].

8 The protocol

A neural network learns the signature (power consumption and electromagnetic analysis) of an instruction, and then recognize it later automatically.

We have to store hundreds traces for each instructions for a processor. So to be able to identify the instruction, we first set a pad to one, to change the power consumption, and then execute the instruction. To reduce the influence of the preceding instruction, we insert a ”nop” instruction before the instruction we want to analyze. We used this instruction because we noticed the influence of the ”nop” was none on the instruction just after. We keep the trace of the instruction (signature) and then repeat the procedure with the electromagnetic field.

So, at the end, we have hundreds signatures for each instruction. Some instructions are more complex than others (one or two parameters). We store several hundreds signatures. We just change one parameter (address of the instruction, data manipulated or address of the data manipulated) to be able to fix a large part of the signature. Once the total set of signatures per instruction is presented to the neural network, it is possible to class them. Each signature defines a zone in a space with its parameters, and the neural network determines the centroid of this area.

Then when you present the power signature or the electromagnetic signature of an instruction to the neural network, it is able to recognize it and to give you the class of the instruction.

9 A practical case: reversing a code

As we did for the correlation, we built two networks, one for the four clock cycles instructions and the second for the rest. Using same detector as before, with in input the shape of an average signature obtained for all 35 instructions, it is possible to isolate the patterns to be presented to the neural network. In the case of our processor, we succeeded in obtaining a reverse engineering of the code and the Hamming weights concerned in 93% of the cases. Figure 7 represents the measured trace with the values obtained at output of the neural network.

The same technique can be used to attack pin codes. Once a network has learned traces of wrong pin code comparison it is able to characterize the difference between the proposed pin code and the right one. We managed to defeat a pin code comparison on a old GSM phone card. It is also very interest-

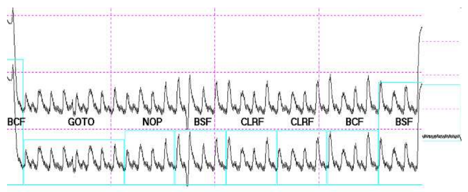

Figure 7: A power trace and the comments after code recognition

ing to notice that some PIN code comparison are still sensible to the timing attack. As the bytes are compared in a given order, and not a random one, it is possible to recover watching the answer time the PIN code. Of course, a very simple countermeasure is to randomize the comparison or to set a constant time before to answer.

In some smart cards, the processor manufacturer has hidden instructions to obtain a specific task. It is well known that some governmental agencies asked Digital to add/remove some instructions to the Alpha processor (Hamming weight). It is very difficult to recover these instructions. So with our method, when instructions appear in the power or EM traces, if they were never presented to the network before, the correlation between the form indicated by the network [Kohonen2] and the treated data shows that the network never met such an instruction in the past. The probability of a correct guess is low for all the outputs.

10 Work in progress

In the future we will focus on the signal processing part. Actually the acquisition chain (acquisition at 125 MHz 12 bits resolution) is good enough and we have to investigate new neural network to improve the selection of each instruction. We decided to start with neural networks based on the K-Nearest Neighbors. Anyway, it does not seem to be the unique solution and a Multi Layer Perceptron sounds quite nice too. Of course a QV based on Voronoi diagram can give results. But we have to test and select new criteria for the network (Manhattan distance). It may be possible to explain completely the influence of the pipeline using a source separator. The representation of the data is very important too, we have to avoid synchronization problems using a wavelet transform or modifying the architecture. We're actually testing a commercial crypto-processor with our network.

11 Conclusion

This article presents a use of traditional techniques of correlation and SOM to find the instructions executed by a very simple processor with only 35 instructions. In order to improve the signal noise ratio of the treated data, the electric field makes it possible under certain conditions to find the exact values of the handled data. Obviously it looks more like an academic case, than a directly usable attack. Indeed the processors for smart cards contain countermeasures, slowing down or preventing such attacks. However it is important to notice that the attacks on side channels will go on increasing, because the possibilities to correlate the data are multiple, and signal processing makes it possible to increase the effectiveness of the attacks.

12 Acknowledgments

We want to thank Cedric Volant for his help.

References

[Kocher] P. Kocher, J. Jaffe and B. Jun, *Differential Power Analysis*, In M. Wiener, editor, Advances in Cryptology - CRYPTO'99, vol. 1666 of Lecture Notes in Computer Science, pp. 388-397,Springer-Verlag, 1999. Also available at: http://www.cryptography.com/dpa/Dpa.pdf.

[Gandolfi] K. Gandolfi, C. Mourtel and F. Olivier, *Electromagnetic analysis : con-*

crete results, In Ko, Naccache, Paar editor, Cryptographic Hardware and Embedded Systems, vol 2162 of Lecture Notes in Computer Science, pp. 251-261, Springer-Verlag, 2001.

[Quisquater] J.-J Quisquater and D. Samyde, *ElectroMagnetic Analysis (EMA) Measures and Counter-Measures for Smart Cards*, in I. Attali and T. Jensen, editors, E-Smart Smartcard Programming and Security,, vol. 2140 of Lecture Notes in Computer Science, pp. 200-210, Springer-Verlag 2001.

[Boneh] D.Boneh, R.A. Demillo, and R. J. Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*, in Proc. of Advances in Cryptology-Eurocrypt'97, Springer-Verlag, 1997, pp. 37-51.

[Anderson] R.Anderson, M.Kuhn, *Tamper resistance - A Cautionary Note*, Proc. of the Second USENIX Workshop on Electronic Commerce, USENIX Association, 1996.

[Messerges] T. Messerges and E .Dabbish, *Investigations of power analysis attacks on smartcards*, In Proc. of the USENIX Workshop on Smartcard Technology (Smartcard'99). USENIX Association, 1999.

[Fahn] P. N. Fahn and P. K. Pearson, *IPA : A new class of power attacks*, Proc. of CHES'99, editors C. K. Ko and C. Paar, Lecture Notes in Computer Science, vol. 1717, Springer-Verlag, pp. 173-186, 1999

[Kommerling] O. Kommerling and M. Kuhn, *Design principles for tamper-resistant smartcard processors*, In Proc. of the USENIX Workshop on Smartcard Technology (Smarcard'99), pp. 9-20. USENIX Association, 1999

[Coron] J-S. Coron, P. Kocher, and D. Naccache, *Statistics and Secret Leakage*, Financial Cryptography 2000 (FC'00), Lecture Notes in Computer Science, Springer-Verlag.

[Kuhn] M. Kuhn and R. Anderson, *Soft tempest: Hidden data transmission using electromagnetic emanations*, In D. Aucsmith, editor, Information Hiding, vol 1525 of Lecture Notes in Computer Science, pp 124-142. Springer-Verlag, 1998

[Biham] E. Biham, and A. Shamir, *Power Analysis of the Key Scheduling of the AES Canditates*, in Second Advanced Encryption Standard Canditate Conference, Rome, March 1999

[Kelsey] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, *Side Channel Cryptanalysis of Product Ciphers*, in Proc. of ESORICS'98, Springer-Verlag, September 1998, pp. 97-110.

[Kohonen1] T. Kohonen, *Self-Organizing Maps*, Third Edition, in Information Science, Springer-Verlag, 2001.

[Kohonen2] T. Kohonen, *The self-organizing map*, Proceedings of the IEEE 78, pp. 1464-1480, 1990.