ElectroMagnetic Analysis (EMA) of Software AES on Java Mobile Phones

Driss Aboulkassimi, Michel Agoyan, Laurent Freund, Jacques Jean-Alain Fournier, Bruno Robisson, Assia Tria

ElectroMagnetic Analysis (EMA) of Software AES on Java Mobile Phones

Driss Aboulkassimi*, Michel Agoyan†*, Laurent Freund*, Jacques Fournier†*, Bruno Robisson†* and Assia Tria†*

Systèmes et Architectures Sécurisés
* ENSMSE
Email: last@emse.fr
†CEA-LETI
Email: firstname.lastname@cea.fr
Site CMP Georges Charpak,
880 Route de Mimet,13541 Gardanne, France.

***Abstract*—Smartphones, whose market share has increased by 54% between 2009 and 2010, is one of the favored platform for "Convergence Computing". Convergence Computing is a technology in which a single device can provide various services without any restrictions from external devices or networks. Today, smartphones as convergent single device have diverse functions and features such as calling, Internet surfing, game playing, banking, storage of personal and professional data, etc. Some of these use encryption algorithms such as AES (Advanced Encryption Standard). For example, this algorithm is used to authenticate server protocols or to encrypt confidential information. This paper shows that an Electromagnetic Analysis (EMA) on AES is possible on a Java mobile phone to extract secret keys. The latter can then be used for forensic purposes or to recover encrypted data stored in the device. Experiments involving two successful approaches are described and compared : Spectral Density based Approach (SDA) and Template based Resynchronisation Approach (TRA).**

I. Introduction and Previous Research

Smartphones are miniature computers embedding advanced software enabling them to execute very complex applications. To ensure the authentication and encryption of data within some of these applications, symmetric encryption algorithms, like the Advanced Encryption Standard (AES [1]), use secret keys stored in the smartphone. The recovery of such secret keys can be useful for forensic and data recovery purposes. Several invasive, semi-invasive and non-invasive attacks are possible to extract them.

An example of an invasive attack is the memory dump. This technique first consists in physically extracting the Flash memory [2] and reading the memory content by using a Flash memory chip programmer. One of the disadvantages is that this can be destructive. This technique also requires a software driver and the right memory content interpretation [3]. Neither is trivial when data-sheets of the Flash memory are not available.

An example of a non-invasive approach is the exploitation of the JTAG port [4]. However, in the latest generation of mobile phones, manufacturers deactivate the JTAG port to avoid this. Another non-invasive technique is called "side channel attack". It exploits the fact that some physical values such as the power consumption, the electromagnetic radiation or the duration of computation of the chip depends on its internal computations [5]. It is of particular concern since it does not destroy the physical integrity of the circuit and it can be quickly mounted with cheap equipment, if several hypothesis are met. The main hypothesis is that the attacker has to be able to measure several times (in practice, from hundreds to millions), the same cryptographic elementary operation with different operands. These measurements have also to be done in the same conditions. In particular, the elementary operations have to be performed at the same time. Examples of extraction of information by using the electromagnetic channel (also called Electro-Magnetic Analysis or EMA) have been reported on a smart card in [6] and on a Java-based PDA (Personal Digital Assistant) in [7].

This article describes and compares two techniques used to analyze software implementations of the AES running on a mobile phone. It is, to our best knowledge, the first successful side-channel attack on a up-to-date mobile phone. We have faced two main difficulties. The first one is the choice of the experimental setup which appeared to be slightly different from those classically described in the literature. The second one is due to the complexity of the smartphone. As the AES is running on Java Virtual Machine (JVM), which is a complex software, the elementary operations of the AES were not realized at the same time (i.e. they were not synchronized). The main hypothesis of side channel attack, mentioned above, was not met. In order to overcome this difficulty, two techniques have been proposed and compared: Spectral Density based Approach (SDA) and Template based Resynchronisation Approach (TRA).

In the first section the AES and the EMA used are described. In the second one, the Java platform on mobile phones is presented in order to explain in which conditions the analysis can be done. Next, the experimental set-up, technical difficulties met and proposed solutions are explained. After that, the SDA and TRA are exposed and compared. Finally, two AES

implementations are analyzed and results are presented.

II. Correlation Electromagnetic attack on AES

A. Advanced Encryption Standard

AES is an algorithm that performs message encryption by data blocks of 128 bits as input and output using a key size of 128, 192 or 256 bits respectively in 10, 12 or 14 rounds according to the size of the key. The algorithm includes two separate processes: one for the key scheduling to derive the round keys from the initial secret key and the second one for data encryption. Decryption is also divided into two separated processes: the first one for the inverse key scheduling and the second one is for the data decryption. In the initial round of AES-128 [1], the algorithm uses the secret key as the round key, but for each following round, the corresponding round key is computed from the previous one. Figure 1 shows the different operations of the AES-128 algorithm.

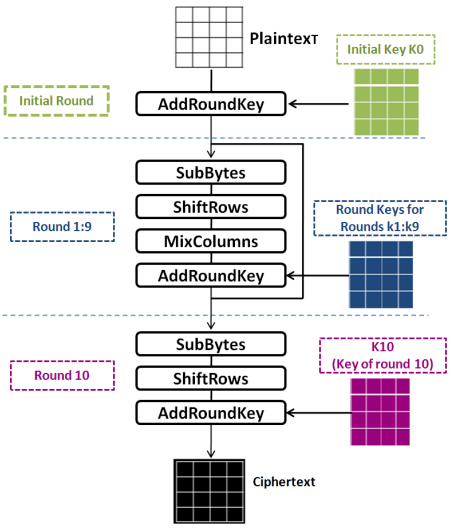

Fig. 1. AES algorithm

B. Correlation ElectroMagnetic Analysis

Correlation ElectroMagnetic Analysis (CEMA) is an EMA technique based on the statistical analysis of the correlation between EM measured during an AES computation and the data processed (secret key dependant). A strategy for Correlation Electromagnetic Analysis would consist in the following steps :

- **step1**. Measure the EM emanation. Let d, a set of D different plain texts for encryption d=(d_1 ,...,d_D), for each d_i is associated an EM curve t_i=($t_{i,1}$,..., $t_{i,T}$) of T points.
- **step2** Choose intermediate result of the executed algorithm. f(d,k) where k is a small part of the key (also called

  guesses key). Let K the set of all the possible values of k.
- **step3** Compute some intermediate values $v_{i,j}$=f(d_i,k_j) for *i*=1,...,D and *j*=1,...,K
- **step4** Map intermediate values to theoretical EM values. A consumption model is used during this step such as Hamming-distance or Hamming-weight models [8].
- **step5** Compare these theoretical consumption values with those measured.

The result is a matrix R of size K×T. The line index of the highest values of the R is the index of the key actually used. The straightforward place for an attacker to find the secret key is during the first round [9]. Therefore, the intermediate values used in step3 are computed according to the SubBytes outputs.

In the next section the Java implementations of the AES and how to setup a CEMA attack are described.

III. CEMA on AES implemented on a Java mobile phone

This paragraph first describes the technical difficulties and solutions for the experimental set-up in order to do our EMA. Second, it shows that the attack is possible on two kind of AES software implementations : the first one is our own Java implementation, and the second one uses a Java cryptographic library (the open source Bouncy Castle cryptographic library [10]). Like for smartcards, EMA on a mobile phone requires EM curves acquisition but there are several difficulties related to the experimental set-up :

- Architecture of the software AES implementation.
- Choice of the EM probe.
- Physically accessing the phone's chip deep inside the mobile phone.
- Generating a trigger signal needed to synchronize the firmware on the mobile phone with our acquisition platform.
- Software set-up for automatic acquisitions of EM traces.

All these parts are described in details in the next sections.

A. AES implementation

Lots of AES implementations exist but this paper focuses on the Java Platform, Micro Edition (Java ME) and the Bouncy Castle library software implementations. To develop a cryptographic Java ME application, developers have the choice among :

- Developing their own implementation according to the cryptographic algorithm specifications.
- Using a commercial or an open source implementation : for example, the Bouncy Castle library.
- Using the manufacturer's implementation for example JSR 177 [11].

The first one requires development but without the help of an external commercial library. Consequently, neither a virus, nor a Trojan is possible. The second one has the advantage of using all the knowledge of a developer's company or community. But royalties must sometimes be paid or open source licence rules must be respected. The last one seems

[1] AES-128 refers to the AES using keys of 128 bits

to be the best solution in terms of performance and security aspects because the manufacturer is responsible for the implementation. The JSR 177 is today only deployed on very few mobile phones.

This paper only focuses on the first and second choices, targeted for a 32 bit processor. In order to optimize the code size and the performances of our own AES implementation, we choose to combine the SubBytes and Shiftrows operations in one operation, implemented as a lookup table in 8×32 bits. In the Bouncy Castle library, the "Fast algorithm" version of the AES has been chosen. In this version, three of the AES operations (SubBytes, ShiftRows and MixColumns) are grouped in one operation, also implemented as a lookup table in 8×32 bits.

B. Experiment set-up

1) Trigger signal: The second technical issue is the generation of the trigger signal. It must be sent just before starting AES encryption. This problem is not as trivial as it seems. Indeed all Java ME applications are executed in a SandBox which forbids direct input/output accesses. The only accessible inputs/outputs on a mobile phone are the screen, loudspeaker and connections like http/SMS/Bluetooth and file access. The solution which guarantees an acceptable response delay is the file access on the MicroSD card. For that, a special extension for the MicroSD's connector has been manufactured to facilitate the access to the MicroSD's PINS (Figure 2). Whenever an access command is sent, some microSD PINS pass to high state. Our tests showed that in order to obtain a trigger signal, the access command must be a simple open file instruction. PIN 2 corresponding to card detection must be connected to the scope acting as the trigger signal).

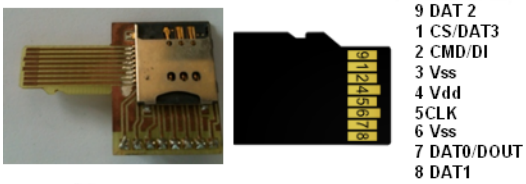

Fig. 2. MicroSD card pinouts

2) Acquisition sequence: The acquisition set-up is illustrated in Figure 3. It consists of an EM probe, an oscilloscope, the trigger mechanism described above and a PC. The Table I sums up the characteristics of these equipments. The acquisition of the EM curves is performed as follows. The mobile phone executes in a stand-alone mode, a large numbers (typically hundreds) of AES encryptions. At the beginning of each encryption, a signal is triggered as explained in the paragraph III-B1. This signal launches the oscilloscope which acquires the EM curves through the EM probe. A sleep time is introduced between two encryptions. This sleep time enables the oscilloscope to send the EM curves to the PC which stores them.

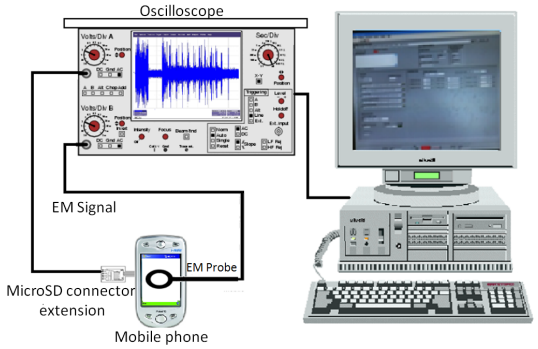

Fig. 3. Experimental set-up

TABLE I
ACQUISITION CONDITIONS AND EQUIPMENTS CHARACTERISTICS

Equipment	characteristics
Target Circuit	RISC Processor, 32bits clock frequency:370Mhz
Oscilloscope	Lecroy 3Ghz, resolution 200 s/div sampling frequency 1GSamples/s
PC	Xeon 2.67 Ghz, RAM 12Go
Soft scope driving	Labview interface Ethernet connection with oscilloscope
plain texts to encrypt	2000 random plain text changing only one of 16 the bytes

Table I, sums up characteristics and parameters of equipments and conditions used during the acquisition phase. The 16 bytes of the text to be encrypted are chosen by randomly changing only 1 byte, according to the attacked secret key byte. Other bytes are kept fixed in order to reduce measurement noise.

3) Choice of EM probe: The choice of the probe is crucial to perform successful EMA. During our measurements two probes are preliminary compared : a homemade one (composed of a solenoid coil of about ten loops of 1 cm diameter) and a commercial one (30MHz-30GHz bandwidth) and as shown in Figure 4, the second one was the best candidate. In fact, an EM characterization demonstrated that a homemade probe is unable to get an exploitable signal in high frequency. It could be explained by misappropriated impedance adaptation and lower probe sensitivity. Performing an impedance adaptation could take a lot of time and the result is not guaranteed. Moreover it's noticeable, that in Figure 4a the noise is greater than in Figure 4b where the ten rounds of the AES are clearly identifiable.

C. EM curves interpretation

Compared to "compiled programs" the execution of an interpreted program (for example Java) needs a virtual machine. This software architecture could introduce a lot of uncontrollable phenomena.

First, the "Garbage Collector" has been identified as one of these phenomena. The garbage collector consists in automatically cleaning the memory. The problem is that this process is launched in an uncontrolled way from a user point of view. Therefore, some EM curves are not exploitable (like

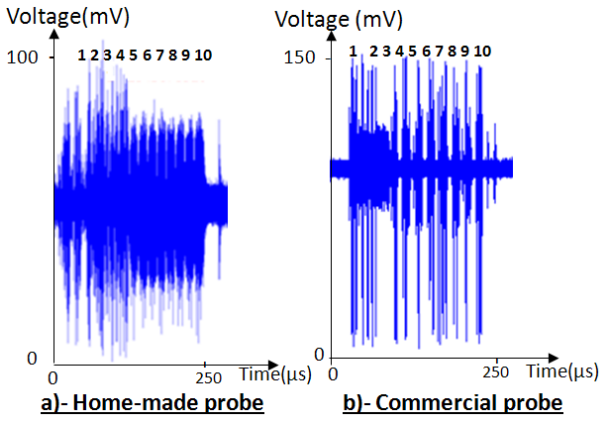

Fig. 4. EM curves captured by commercial and homemade probe

in Figure 5) because the signal is indistinguishable from the noise. As the garbage collection consumes power and thus increases the electromagnetic radiation of the chip, the curves where the "Garbage Collector" is triggered are discriminated simply by computing their temporal average. The average of such curves appears to be 20% higher than the curves where the collector is not triggered.

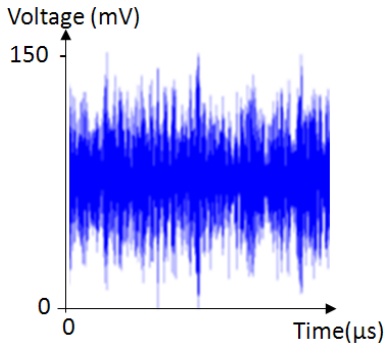

Fig. 5. Curve acquired while garbage collector was active : no AES visible compared to 4b

Second, the"Just-In-Time-Compiler" was identified as a possible problem. Indeed, the "Just In Time Compiler" is an optimization executed by the virtual machine to speed-up the multiple execution of a set of instructions. It consists in compiling the instructions on-the-fly during their first execution. This phenomenon is visible in Figure 4b where the first round takes more time than the following ones. These two issues introduce EM curve's misalignment. The temporal shift among EM curves varies between 0 to 110ms and may impact the EMA.

In order to find a workaround to the EM curve's misalignment issue a timing synchronization methodology based on the detection of the beginning of the first round was tried. The statistical analysis failed due to temporal shift between instructions within the round itself. To solve this problem two successful solutions are described namely SDA and TRA.

D. Electromagnetic Analysis with Spectral Density based Approach

The misalignment could also be due to random delays that could be introduced as a countermeasure in AES hardware and software implementations. Several techniques in signal processing exist and could be exploited, such as in [12] [13]. One of the synchronization techniques based on the frequency approach is introduced in [14]. The authors show the efficiency of DEMA on a very high speed embedded system. This technique is based on the fact that the Power Spectrum Density (PSD) of a "shifted signal" and the PSD of a "not shifted signal" are the same.

In [15], the authors show that even if random delays are introduced, the CEMA attack in the frequency domain is still efficient on a program executed without a virtual machine on a processor running at less than 12 MHz. Unlike previous researches, our paper shows the efficiency of the CEMA attack on a AES Java program executed on a virtual machine on a high speed circuit (400MHz). [16] exploits the simulation of the power consumption by using CPA (Correlation Power Analysis). Their analysis in frequency domain allowed them to find the secret key.

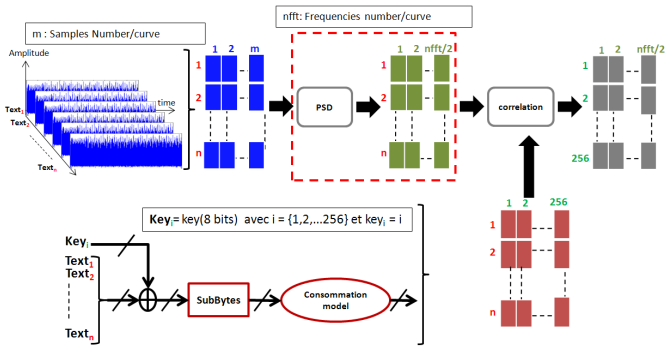

Fig. 6. CEMA in frequency domain

We performed such a statistical analysis in the frequency domain on our own implementation of the AES. As represented in Figure 6, it consists in applying the algorithm in II-B not on the raw EM curves but on their PSD. More precisely, we first remove the curves which includes the garbage collection step. Second, the PSD of the remaining curves (1500 curves) is computed ; next, for the 256 key guesses, the power consumption model (in this case, the Hamming Weight) for the intermediate values (in this case, SubByte output) is also computed. Finally, statistical analysis by computing the Pearson's correlation factor between PSD and the Hamming Weights has been applied. The result of the correlation is represented in Figure 7 for one key-byte where the correlation factor of the right key guess is distinguishable from the others. Next, acquisition and computing steps are iterated to find the remaining key bytes.

With this result, the efficiency of the CEMA attack has been shown on our AES J2ME program executed on a virtual machine on a recent mobile phone (400MHz).

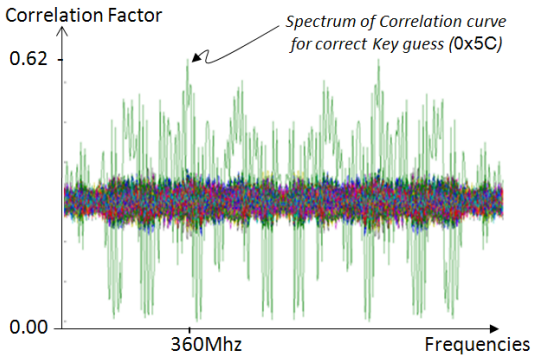

Fig. 7. Correlation result for one key Byte in the frequency domain

E. ElectroMagnetic Analysis using Template Resynchronisation Approach

In our experiments, it has been observed that the EM signature of all SubBytes curves of our AES implementation are very similar. It can be explained by the fact that the SubBytes are implemented as a Look-Up-Table (LUT), i.e. an access in an array and that this access emits a characteristic EM signal (i.e. a signature). We use this property in order to speed up the attack. The proposed approach consists, first in extracting the signatures of the SubByte operation in the EM signal (as represented in Figure 8) and them performing the statistical analysis described in II-B in temporal domain on these extracted signatures. More precisely, the following pseudo-code describes the proposed approach:

- first a code of an LUT access has been developed. Its signature has been defined as the reference pattern (or "template").
- by using a sliding window technique correlation, this pattern has been researched in all curves.
- finally the CEMA is applied on the identified regions.

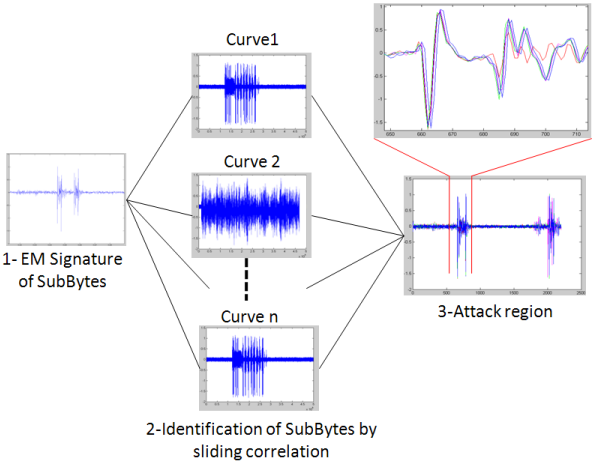

Fig. 8. CEMA by SuBbytes identification

Thanks to this technique, each byte of the key has been recovered with only 256 EM curves (Figure 9). The correlation factor is about 72%. If more curves are used the correct guess key is more distinguishable (Figure 9).

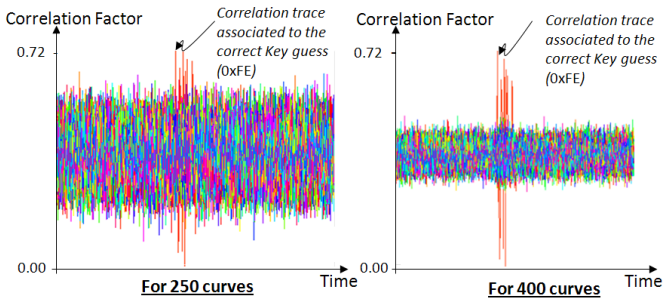

Fig. 9. Correlation result for one key Byte

TABLE II
COMPARISON BETWEEN THE TWO PROPOSED APPROACHES

	SDA	TRA
Number of curves	20800	4096
Sorting curves	yes	no
Number of samples	1 Million	600
computing time	28h	1h

F. Comparison

With the previously described results, it has been proven that SDA and TRA are efficient techniques to manage desynchronisation introduced by a mobile phone's JVM during an EMA. The differences between both methodologies are summarized in Table II. As shown in this table, the SDA needs to select curves without "Garbage Collector" while the selection is automatically realized by the SubBytes operation identification technique of the TRA. An important point to notice is that the first one requires four times more curves than the second one. On the opposite, the second one needs to build a template before performing the attack. The real disadvantage in the first one is the number of samples (millions) compared to the second one (hundreds). Consequently more computer resources are necessary to perform the DSP and to apply the CEMA.

It has been shown that CEMA on our AES implementation on a mobile phone is also possible despite "irregularities" introduced by the JVM. By comparing both proposed approaches, the fastest has been identified. The next interesting challenge was to attack an AES cryptographic library using TRA.

G. EMA on Bouncy Castle AES for mobile phone

1) BouncyCastle implementation: Bouncy Castle is an open source lightweight library used by Java applications which need cryptography. It provides support for standards such as Transport Layer Security, Public Key Infrastructure and Certificate Management Protocol. This paper treats the AES implementation part.

To begin, a first acquisition was made. Visually, only 6 "patterns" were identifiable, compared to our implementation (Figure 10).

In fact, the analysis of the code shows that AES rounds are grouped in pairs. That explains the EM curve shape. The

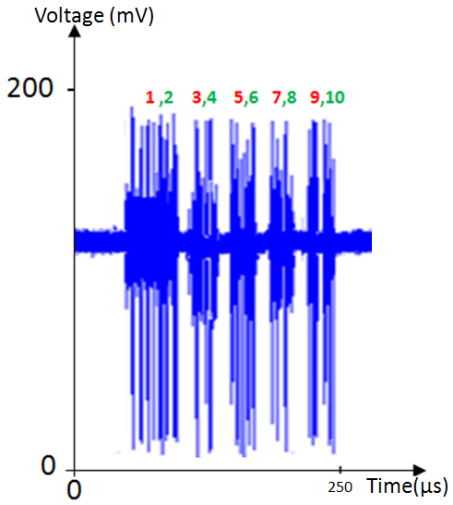

Fig. 10. EM curve corresponding to AES implementation of Bouncy Castle

challenge is to find the key without any knowledge of the source code.

2) Methodology and attack results: Our hypothesis was that the implementation of the SubBytes could be the same as on a 32 bit processor architecture, i.e. a 8×32 LUT. Therefore the template previously defined for our implementation could be used for the identification of the Bouncy Castle's SubBytes function operation. Processing has been made in this manner and it has been discovered that the CEMA succeeds with only 250 curves (for each byte of key) with a correlation factor of 77% (Figure 11).

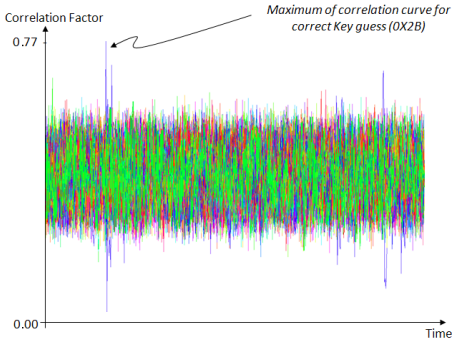

Fig. 11. Correlation result for Bouncy Castle AES (for one key byte)

This study demonstrates that an EM attack is possible without knowing the source code of an external library. This kind of attack could be dangerous because it could be generalized to all AES implementations on a 32 bit processor where the SubBytes operation is implemented by a 8x32 LUT.

IV. Conclusion

The security of the mobile phone becomes a crucial point because more and more applications manipulate private data and run sensitive applications putting into play cryptographic algorithms with secret keys. Retrieving such secret keys can be vital in some forensic procedures or for the recovery of encrypted data. In this article, two methodologies have been proposed and compared to do CEMA : a Spectral Density based Approach and Template based Resynchronisation Approach. These methodologies have been tested on two different implementations of an AES. We show that both solve the misalignment problem. It also has been shown that the TRA can be transposed to a legacy cryptographic library and can probably be applied to other Java platforms. With the TRA, one byte of the key has been found in 1 hour and by analyzing only 250 curves. This study highlight the fact that Java developers have to use secure cryptographic libraries embedding countermeasures against EMA. A secure hardware cryptographic implementation is also possible to secure mobile phones.

Acknowledgment

The experiments were done on the MicroPackS™ platform and funded by the Pôle System@tic's FUI project TISPHANIE.

References

[1] National Institute of Standards and Technology (NIST), *Announcing the advanced encryption standard (AES)*, Federal Information Processing Standards Publication, vol. 197, 2001.
[2] Marcel Breeuwsma, Martien de Jongh, Coert Klaver, Ronald van der Knijff and Mark Roeloffs, *Forensic Data Recovery from Flash Memory*, Small Scale Digital Device Forensic Journal, Vol. 1, No. 1, 2007.
[3] Kim, K., Hong, D., Chung, K., Ryou, *Data Acquisition from Cell Phone using Logical Approach*, Proceedings of World Academy of Science, Engineering and Technology. Vol. 26. December 2007.
[4] Svein Y. Willassen, *Forensic analysis of mobile phone internal memory* In IFIP Int. Conf. Digital Forensics, pages 191204, 2005.
[5] P.C. Kocher, *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems*, Advances in Cryptology, Crypto 96, LNCS 1109, N. Koblitz, Ed., Springer-Verlag, pp. 10411, 31996.
[6] K. Gandolfi, C. Mourtel and F. Olivier, *Electromagnetic analysis: concrete result*, In Ko, Naccache, Paar editors, Cryptographic Hardware and Embedded Systems, vol. 2162 of Lecture Notes in Computer Science, pp. 251-261, Springer-Verlag, 2001.
[7] C. Gebotys,S. Ho,A. Tiu. *EM Analysis of Rijndael and ECC on a PDA*, Technical Report:CACR 2005-13, Dept of Electrical and Computer Engineering, 2005.
[8] Brier, E., Clavier, C., Olivier, F. *Correlation power analysis with a leakage model.* In Proceedings of the Cryptographic Hardware and Embedded Systems- CHES 2004. 6th International Workshop, (Cambridge, MA, USA, 2004), Springer Verlag, 16-29.
[9] Mangard, S., Oswald, E. and Poop, T.,*DPA Book*
[10] www.bouncycastle.org
[11] jcp.org/aboutJava/communityprocess/final/jsr177/index.html
[12] J. van Woudenberg, M. Witteman, and B. Bakker. *Improving differential power analysis by elastic alignment* , www.riscure.com/fileadmin/images/Docs/elastic_paper.pdf
[13] N. Homma, S. Nagashima, Y. Imai, T. Aoki, and A. Satoh, *High-resolution side-channel attack using phase-based waveform matching.* CHES 2006, LNCS, vol.4249, pp.187200, May 2006.
[14] C.C. Tiu, *A new frequency-based side channel attack for embedded Systems.* MS thesis, Dept. of Electrical and Computer Eng., Univ. of Waterloo, 2005.
[15] Zhang, P., Deng, G., Zhao, Q., and Chen, K. *EM Frequency Domain Correlation Analysis on Cipher Chips.* In Proceedings of the 2009 First IEEE international Conference on information Science and Engineering, December 2009.
[16] Schimmel, O., Duplys, P., Boehl, E., Hayek, J.,Bosch, R., and Rosenstiel, W. *Correlation power analysis in frequency domain.* In COSADE 2010 First International Workshop on Constructive SideChannel Analysis and Secure Design, 2010