




A Framework for Acquiring and Analyzing Traces from Cryptographic Devices

Alfonso Blanco Blanco¹, Jose María de Fuentes², Lorena González-Manzano²,
Luis Hernández Encinas¹ , Agustín Martín Muñoz¹,
José Luis Rodrigo Oliva¹, and J. Ignacio Sánchez García¹

¹ Departamento de Tecnologías de la Información y las Comunicaciones,
Instituto de Tecnologías Físicas y de la Información,
Consejo Superior de Investigaciones Científicas, Madrid, Spain
{alfonso,luis,agustin,joseluis.rodrido,nacho.sanchez}@iec.csic.es

² Departamento de Ciencias de la Computación,
Universidad Carlos III de Madrid, Leganés, Madrid, Spain
{jfuentes,lgmanzan}@inf.uc3m.es

Abstract. We present a Side-Channel Analysis Platform (SCAP) Framework developed to acquire and study the traces derived from a cryptographic device when cryptographic computations are done. The main goal of this work is to develop a tool for performing side-channel attacks against these cryptographic devices. The characteristics of the SCAP Framework are described and a case study with a smartphone is presented.

Keywords: Android · Power consumption · RSA traces
Side-channel attack · Smartphone

1 Introduction

The uses of electronic devices with cryptographic features in order to perform personal and business operations has largely increased worldwide in the last years. In general, most of these uses (identification, payment with credit cards, access to the cloud, browsing, etc.) are relevant enough to ensure important security measures.

The cryptographic community considered that the security of a cryptosystem lied in the strength of the mathematical problem in which it was based on. For instance, the security of the RSA algorithm is based on the difficulty of solving the integer factorization problem [1], or the strength of the elliptic curve cryptography (ECC) is based on the intractability of the discrete logarithm problem over elliptic curves defined on finite fields [2].

Nevertheless, this paradigm has been questioned since the publication in 1996 of a paper by Kocher [3]. Kocher demonstrated that it was possible to break the

security of embedded cryptographic protocols by means of an attack which, instead of trying to solve the underlying mathematical problem, made use of the information obtained from the cryptographic device, i.e., from the fact that the cryptosystem was physically implemented in such device. Kocher showed that measuring the time required to perform private key operations could be useful to find Diffie-Hellman exponents, to factor RSA keys, etc.

Nowadays, the continuous development of device-implemented cryptography [4] is accompanied by an increasing number of physical attacks [5–7].

The information obtained from the implementation of a cryptographic algorithm in a device is related to the execution of the code and can be obtained by measuring the computation time that the algorithm takes to execute [3], the consumption of electric power to execute the process that is running [8], the generation of electromagnetic fields during their computations [9,10], the temperature reached by the chip [11], the noise produced while doing calculations [12], etc. All channels which permit to obtain extra information about the cryptographic processes are called *side channels* and the attacks derived from the information obtained are denoted as *side-channel attacks*. Note that these attacks are passive in the sense that they do not modify the device where computations are done; hence, they are difficult to be detected.

Side-channel attacks consider that the mentioned measurable quantities depend on the instructions, mathematical operations, and the data used by the processor to perform its cryptographic computations. In this way, if the implementation is not sufficiently protected it is possible to obtain information related to the keys through these side channels.

On the contrary, when the attacker provokes a fault or malfunctioning in the device due to the alteration of its normal execution such as modifying the temperatures accepted by the device, triggering a laser that alters the memory contents or the execution flow of an algorithm, etc. [13], then we are considering *fault-injection attacks*.

In this work we present SCAP (Side-Channel Analysis Platform) framework, a framework to develop a tool for acquiring and studying the obtained traces when a cryptographic device is carrying out cryptographic computations. To illustrate the benefits of our framework we have applied it to the acquisition of the power consumption traces of an implementation of the RSA cryptosystem over a smartphone. We have selected the RSA cryptosystem because it is the most extended system nowadays. Nevertheless, the framework also includes an implementation of the ElGamal (EG) cryptosystem. In a general way, the framework could be easily extendable to any other cryptosystem as ECC, for example. Our framework has been developed in LabViewTM.

The rest of this paper is organized as follows. In Sect. 2, a short review about side-channel attacks is presented. Section 3 presents our Framework, including a description of the smartphone interface and the used hardware and software. In Sect. 4, the experimental results obtained with the mentioned hardware and software are shown. Finally, Sect. 5 includes the conclusions of the work and possible future work.

2 Side-Channel Attacks

As we have mentioned, problems related to the security of implementations arise due to the existence of side channels on the device from which it is possible to obtain sensitive information, analyzing either the behavior of the software or the hardware, and inducing faults in the behavior of the circuit to deduce information about the keys.

These attacks on physical devices are more specific than the classic ones since they are carried out depending on the algorithm implementation, the chip architecture, etc. They are classified as invasive, semi-invasive or non-invasive [14, 15], depending on whether the device is manipulated or only the available information is used. Another classification of the attacks is: active or passive, depending on whether they manipulate the device or only observe its behavior, respectively.

Finally, it is noteworthy that, as in traditional attacks, it is assumed that the Kerckhoffs principle [16] is verified, i.e., the starting point is that the attacker has access to the device, he knows the cryptographic algorithm implemented in the chip, the characteristics of that implementation, and he can execute the protocol with the parameters he estimates appropriate as many times as desired.

The main types of attacks against physical devices are summarized below.

2.1 Timing Analysis

The attacks for *timing analysis* try to obtain information about a cryptographic protocol by measuring the time that the attacked physical device takes in performing the operations of the algorithm being attacked [3].

For example, in the case of the RSA cryptosystem, the attacker wants to obtain the private key while executing the operation of the modular exponentiation. This computation is carried out by the squaring and multiplying algorithm. The initial hypothesis is that the time required for a multiplication is constant, but if a modular reduction must be performed due to the fact that the multiplication is greater than the module, then this new operation implies an increase in the execution time. This way, it is possible to obtain some information about the sizes of the numbers considered in each step of the algorithm.

2.2 Power Analysis

In some cases, certain information can be extracted by measuring the power consumption of the microprocessors. This consumption can be closely related, for example, to the number of bits that change in memory or register. Thus, an attacker can take advantage of this feature to try to guess a secret value used in a cryptographic operation by observing the power consumption trace, for example.

There are several attack methods related to this side channel. The most simple methods are the *Simple Power Analysis* (SPA) attacks. These attacks use the power consumption traces measured when the cryptographic device is working. Traces are obtained by a digital oscilloscope that measures the voltage

drop in a resistor that is connected to the power supply of the device. From the measurements of a trace (or a few traces), the attacker will try to obtain some information about the secret key used in the implemented cryptosystem.

When the obtained signal is weak or if the relationship between the secret key and the consumed power is not clear, SPA attacks do not give enough information to break the algorithm. In these cases, a new type of attack, which uses statistical techniques, is considered: *Differential Power Analysis* (DPA). DPA uses a lot of traces of power consumption and requires a synchronization and alignment of the measured traces. Then, a statistical analysis can be made between the values of the power consumed along the execution of the algorithm with the values of the hypothetical model [17].

In the *Correlation of the Power Attacks* (CPA), the correlation between the measurement of the instantaneous consumed power and the data processed is analyzed [18]. As this correlation is, in general, very small, it is necessary to obtain a large set of measurements in order to have a lot of traces which are compared, by means of correlation coefficients, with the traces from the outputs of a theoretical model of the device.

Finally, the *High-Order Differential Power Analysis* (HODPA) are a generalization of the DPA attacks. In this case, several points of the power trace are used [19].

2.3 Electromagnetic Analysis

Electromagnetic fields emitted by a circuit due to the displacement of charges along the tracks of the metal layers of the circuit can be measured when the transistors switch state [20], which gives rise to the *ElectroMagnetic Attacks* (EMA).

Once these emanations are measured, they are analyzed in a similar way as the power traces, so that they give rise to *Simple ElectroMagnetic Analysis* (SEMA) or *Differential ElectroMagnetic Analysis* (DEMA).

Traditionally, EMAs have been used to attack smart cards, FGPAs and other small devices; nevertheless, attacks against laptops have also been carried out. In [10] a laptop has been attacked by using an antenna of 0.5 m which is connected by a coaxial cable to a low-pass filter and two amplifiers.

2.4 Other Type of Attacks

The sound produced by a device can be used as a side channel as well. They are called *acoustic attacks*. For example, in [12] two attacks are described. In the first attack, the microphone of the smartphone Samsung NOTE II, located at 30 cm, points to the ventilator vents of the notebook and an attack against the secret key used in computations of the notebook is performed. The second one considers a parabolic microphone connected to a laptop in a padded case which attacks to another laptop, located at a distance of 4 m from the first one.

Attacks denoted as *non-invasive physical attack* can be mounted by measuring the fluctuations in the electrical potential of the chassis of a laptop by means

of the grounding or with a conductor cable connected to an input/output port, or even by touching the equipment by hand and measuring the potential of the body [21].

In some situations, attacks can be mounted by using several methods among those discussed above. In this way, it is possible to increase the power of the attack being carried out. For example, by simultaneously combining power consumption and electromagnetic emanations [22].

Next, we include other type of attacks. These attacks are specifically proposed against different implementations of cryptographic primitives on smartphones. In [23], authors have proposed a pre-processing composition to mount a Simple Side-Channel Analysis (SSCA) on RSA and ECC, i.e., an attack that uses one single waveform to uncover a secret key. In particular, they explain how a composition of time-frequency pre-processing manages to extract the relevant information obtained from one signal of an asymmetric cryptographic operation (RSA and ECC) running on an Android system.

On the other hand, electromagnetic emanations of smartphones have been used in [24] to obtain secret keys of public key cryptosystems by means of standard radio equipment in combination with far-field antennas. Moreover, in [25, 26], side-channel resistance of the implementation of the ECDSA signature scheme in Android's standard cryptographic library is studied. Authors show that, for elliptic curves over prime fields, it is possible to recover the secret key very efficiently on smartphones using EMA side channel and lattice reduction techniques.

In [27], it is shown that elliptic-curve cryptography implementations on mobile devices are vulnerable to electromagnetic and power side-channel attacks. Authors prove that full extraction of ECDSA secret signing keys from OpenSSL and CoreBitcoin running on iOS devices, and partial key leakage from OpenSSL running on Android and from iOS's CommonCrypto are possible. The mounted non-intrusive attacks use a magnetic probe placed in the proximity of the device, or a power probe on the phone's USB cable.

3 The SCAP Framework

We have designed a framework, called SCAP (Side-Channel Analysis Platform) Framework, for capturing and studying the traces obtained when a cryptographic device is carrying out cryptographic computations. In order to show the behavior of this framework, we have considered the capture of traces derived from the power consumption when the RSA cryptosystem is running under an Android smartphone. In fact, we have implemented both the RSA and the EG cryptosystems in the smartphone so they are accesible by means of an Android application used as the interface to communicate with the smartphone. We will try to determine if it is possible to obtain side-channel information that allows breaking that implementation when the smartphone is deciphering.

The RSA and EG implementations have been made by using two different libraries: the Spongy Castle, SC, (<https://rtyley.github.io/spongycastle/>) and

the BouncyCastle, BC, libraries (<https://bouncycastle.org>). The specific library can be chosen by the user when he launches the application.

3.1 Overview of the Framework

Side-channel attacks against cryptographic devices or, in general, against a Device Under Test (DUT), involve two types of activities: (1) The creation of a database of traces obtained from the measurement of different types of parameters when the device is computing (time, power consumption, emanations, etc.), and (2) the processing of such data. This processing is done by taking into account the knowledge of the algorithm, the protocol performed in the DUT, and a mixture of observation and intuition together with techniques of signal processing, statistics, etc.

The database is elaborated by collecting informative fragments from the repetition of observations, with similar or different parameters. The files with the obtained data are stored with some extra information about the involved parameters and variables, due to the number of files considered (sometimes more than 100 000 files). So, it is very convenient to automate, as much as possible, the acquisition and storage of data.

Moreover, given the large amount of data considered in each file, it is necessary to use an efficient data model in the sense that it is necessary to get a balance between data volume and the ease of its processing, i.e., the amount of data stored and its format must be adequate in order to avoid the slowing down due to the use of limited memory resources, virtual memory, type of processor, etc. Finally, it is important to consider that side-channel attacks involve the interaction with a set of hardware devices like oscilloscopes, data acquisition cards, etc., and software applications.

Indeed, in our SCAP Framework, we have considered the challenge of developing a database through a double strategy: on the one hand, the creation of a big data pool, with sorted, nominated and indexed files; and on the other hand, the management of these processed files, using database tools and data mining, making use of the parameters and results stored in such files.

To do this, we name the files with prefixes and suffixes generated automatically or manually, so that their names contain the test performed, the date and time of creation, and an extension. The files are stored in a directory tree with several levels. In the first level there are two branches: one branch stores the files with source data, and the other one, with different directories depending on the type of analysis, contains the files with the processed data. If the file size is large and a high efficiency is essential, strictly binary files are used. However, the “TDMS” (Technical Data Management Streaming) format is preferred as it handles variables in a standardized way together with metadata with group names, experiment parameters, etc., and moreover, it includes an implicit indexing system that facilitates subsequent data mining.

Data management is done using data mining tools (DIADEMTM, National Instruments), proprietary applications developed by our group, and small pro-

grammed tools (SQLite); all of them in order to extract the parameters of the data files, create files summary, etc.

3.2 Components of SCAP Framework

SCAP framework is composed of hardware devices and a set of software applications (see Fig. 1). In the following elements of SCAP framework are described.

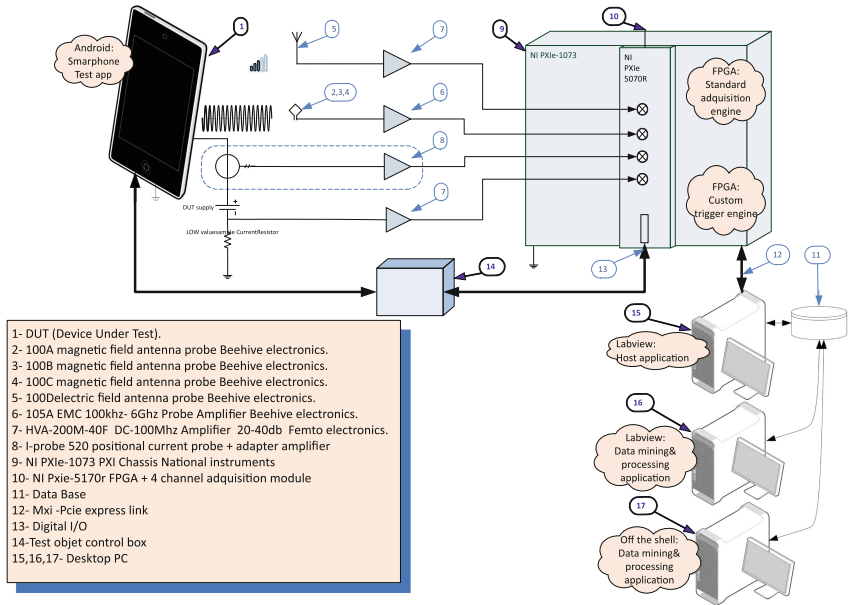


Fig. 1. General scheme of the SCAP Framework

- **DUT (1):** It is the entity to be observed, where possible leakages are produced. In general, DUT contains some type of application or software (Test app), with known characteristics and algorithms for the observation.
- **Test Object Control Box (14):** It produces an excitation in the DUT, adapted to the type of test. The excitation can be generated manually or through some automatically generated vector.
- **Data Acquisition System (9)–(10):** The DUT produces some type of emanations or leakages that are captured by probes. These probes, individually or collectively (2)–(8), provide to the Data Acquisition System analog signals that correspond to emanations of different types: magnetic, electrical or electromagnetic radiation, variations in power consumption, temperature, etc.

These signals are introduced into Data Acquisition System, where an application, developed in Labview™, is installed as a firmware (we will talk about it later) on a hardware equipped with Analog Digital Converters.

The received data sequence is controlled, preprocessed and filtered at low level by a FPGA (Field Programmable Gate Array) with local memory. The FPGA creates a specially selected data stream, which is transferred at high speed, via PCIexpress (12), to the host PC. The PC host processes the data through a host application designed for this purpose, and stores the processed data in a database (11) for sharing and subsequent post-processing.

On the other hand, the FPGA (10) controls via the digital lines, supplied by the Digital IO port (13), different communication protocols for the DUT excitation (vectors or triggers) through the Test Object Control Box (14).

- **Processing Set (15)–(17):** Several applications access to the database to process the attack; they are developed ad hoc in LabviewTM, C, and standard packages off the shell (Matlab, Diadem, etc.). These applications can be used cooperatively, from other workstations (desktop, laptop, etc.), for curve view, data browsing, data minning, and manipulating the information of the files.

The described architecture is flexible, so we are going to explain in more detail, for a particular implementation, some of the main parts.

3.3 Test Object Control Box

There are several ways to produce some kind of excitation to the DUT (in our case a smartphone). In general, the DUT needs to receive, in some way, the command to generate a set of events that produce the possible leakages. A simple way to do this is to develop an app where one can induce this type of situations, for example, by touching the smartphone screen or by an external mechanical push button switch and fixing the smartphone, if many trials are needed.

Another way to execute the command is through the digital IO port interface (see (13) in Fig. 1), which improves the determinism of the test. In our case we execute the push button switch action with an off-the-shelf pulse generator: an Agilent 33220A (see Fig. 2).

The connection of the pulse generator with the DUT can be made by using cheap hardware (Arduino boards, for example) or, as in our case, with the ear-phone connector and a small interface like the selfie stick to activate the camera button.

To detect the trigger that marks the start of the operation to analyze, we must add a suitable routine in the app running on the DUT, and an output line that provides a trigger signal to be used by another hardware (Data acquisition system, Oscilloscope, etc.) that helps to the synchronization of the traces.

3.4 Data Acquisition System

This device performs the acquisition of leakage signals. In general, this functionality is well known and it is typically carried out by using data acquisitions boards inside the PC, and especially by digital oscilloscopes. However, we prefer the use of hardware with high-speed, high-resolution A/D converters, FPGA, and PCIexpress link to PC.

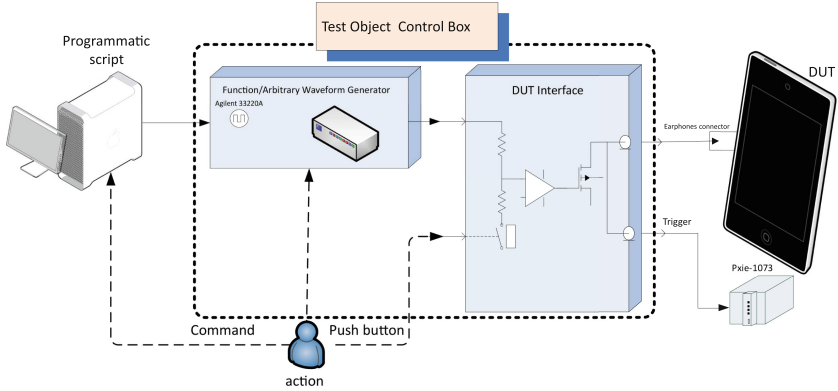


Fig. 2. Test Object Control Box

To study a DUT, fragments of information varying from several thousands of a second to several tens of seconds have to be observed and recorded. However, useful leakage information is usually contained between tenths and tens of milliseconds. In order to have sufficient temporal resolution and to be able to discern the useful part, it is necessary to sample between 40 ms and 250 ms. On the other hand, once the useful area has been determined, and after being filtered, it is possible to repeat the sampling reaching values between 10 ms and 20 ms, i.e., the information can be summarized as 0.001% and 0.010% of the total.

Another question to consider is that different side-channel attacking techniques require generating tracks of data with respect to a well-known temporal source, as accurate as possible, with little jitter with respect to the initial trigger. This is an important problem because the synchronization of traces is necessary. Thus, it is very important to make the collection data as accurately as possible with reference to the same source of temporal coordinates, and to have a flexible, combinable and masked trigger subsystem.

In the designed framework, our Data Acquisition System is constituted by a NI PXIe-1073 Chassis and an NI PXIe-5170R module, manufactured by National InstrumentsTM (see Fig. 3). The NI PXIe-1073 Chassis allows you to feed the modules that are introduced, to synchronize them and to establish a communication link with a PC through an MX interface. Moreover, the NI PXIe-5170R consists of a general purpose reconfigurable multichannel digital oscilloscope whose internal controller consists of a reprogrammable FPGA. It consists of 4 channels of simultaneous acquisition of up to 250 ms/s, 14 bits of vertical resolution, amplifiers and anti-aliasing filters, with inputs programmable, digital outputs, individually or with serial protocols: I2C, SPI, etc.

There are multiple possibilities for synchronizing and exchanging data with other modules communicated with a PC using PXIexpress Gen2 x8 (up to 3.2 Gb/s), which are inserted in a NI PXIe-1073 (up to 200 Mb/s) chassis.

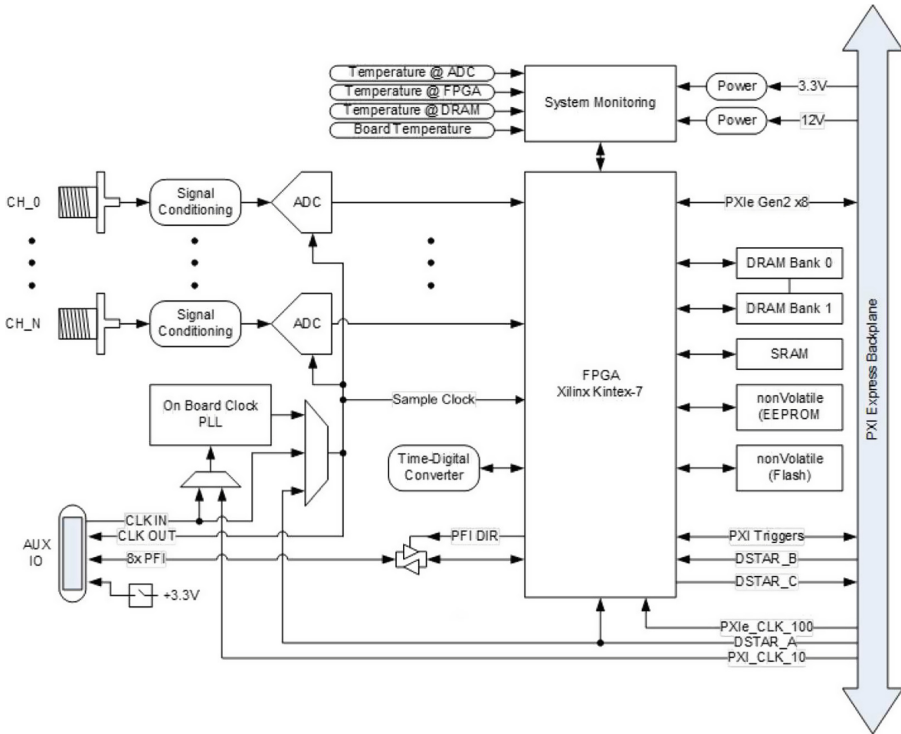


Fig. 3. NI PXIe-5170R block diagram. (Courtesy of National InstrumentsTM)

Regarding the implementation of the Data Acquisition System, the main goal is to obtain a functionality similar to that of a digital oscilloscope, with extended and customized triggering capabilities (for masks and combined multi-shot), a pre-processed (typically filtered and decimated) that, together with the customized trigger, provides a specially selected data flow. This flow will be transferred at high speed, via PCIexpress, to the computer, where it is processed by an ad-hoc application. The flow is stored in a database that allows sharing and post-processing by means of two types of applications: firmware in PXIe-5170R FPGA Oscilloscope Emulation Module, and Host Application for Data Transactions.

Next, we will describe with more detail both applications: firmware and Host, which were made in LabviewTM environment using the XilinxTM ISE and Vivando tools.

Firmware in PXIe-5170R FPGA Oscilloscope Emulation Module. The implementation of the firmware (to be run in the FPGA) is performed to handle all hardware acquisition such as A/D converters, analogue digital filters, sequencers, DMA, etc.

The strategy of mixing two types of libraries is used: a standard realization with LabviewTM Instrument Device Libraries (IDL), and a specific customization of it. This way the emulation of a multichannel oscilloscope with its most characteristic primitives is performed: vertical and horizontal range change, trigger mode (simple, continuous), data storage size, transparent mode of transport of data to the computer memory, and FIFO communications.

The triggers customization allows the detection of events generated by various trigger forms from any channel, with storage, multi-trigger, multichannel combinational logic conditions trigger (And, Or, If then, etc.) and arbitrary mask trigger with pattern and (stored in a memory pattern) trigger tolerance.

These deeply coupled to acquisition process triggering methods allow the detection of an event or anomaly in a particular pulse, with a certain value, with a given waveform, without any need of storing previous data. This fact, together with the possibility of “selective decimation”, constitutes a very important basis for saving data transactions and storage space over long periods of data observation.

Host Application for Data Transactions. This application, developed in LabviewTM, allows sending or receiving commands and data to the acquisition system, the transfer of files to the database and their processing.

The architecture is based on a typical multitasking cue loops producer-consumer, with isolated tasking loops: to the human interface, graphics curves, data acquisition transactions. The graphical interface has an aesthetic of folders with tabs containing functions (see Fig. 4).

Some options available in the host app are: communication port, data acquisition parameters, vertical-horizontal range, conversion rate, decimation, filter (media, average, median, envelope extraction, etc.), data fetch visualization, pre- and post- trigger (size and count), trigger mode (edge, level, channel, condition, combination, mask), Hilbert transform, etc. Moreover, it is possible to change of graphic mode view, colors, interpolation mode, channel in view, etc.

Moreover, graphs can be exported to elaborate reports or documents and save data in different formats for data export purposes.

3.5 Smartphone Interface

In this section we briefly describe the environment where SCAP Framework has been used.

The smartphone (DUT) used has the following characteristics: it is a Samsung, model Galaxy S3 GT-i9300 16 GB. The operating system is an Android 4.4.4 CyanogenMod, version 11-20141115-SNAPSHOT-M12-i9300, the kernel version is 3.0.64-CM-g4ca83ff, Build02@cyanogenmod #1, Fri Nov 14 21:44:13 PST 2014, and the CPU is ARMv7 Processor rev 0 (v7I).

When the user launches the developed app, he must select some possible options (see Fig. 5). The first action is to select the cryptosystem, in this case, RSA. Then, he chooses the library used for the implementation: SC or BC, and



Fig. 4. Screenshot of the host app

next, the user must choose the bitlength of the RSA key, where three options are available: 512, 1024, and 2048 bits.

Once the characteristics of the cryptosystem have been selected, the user selects how long the light of the flash (Time light) will be activated (in ms), the number of times (Loop time) that he wants to run the decryption process, that is, the number of times that the ciphertext will be decrypted with the same key, and the length of the plaintext to be used, which depends on the bitlength of the key. The plaintext is formed by repeating the chain ‘0123456789’ the number of times needed until texts of length 72, 200, 352, 472, or 792 bits, are obtained.

The app is launched by pressing the Start button, which can be activated in two ways: by pressing it with a finger or by means of an external trigger connected to the audio jack (like the system used to make a selfie).

The first time the application is launched, it generates the couple of keys (public and private) and the plaintext with the selected length. Then, it encrypts the plaintext obtaining the ciphertext and stores both the keys and the ciphertext in the smartphone. When the application is launched again, it will verify if such data is stored. If the data exists, it will use it; if not, it will repeat the previous process to generate the keys and the ciphertext for the current options. After these verifications, the decryption process is executed as many times as selected. Note that the flash is turned on before and after this process is carried out. The light of the flash is used as a signal or trigger.

The developed app turns on the flash of the camera in each encryption a unit of time, two units of time in the decryption and finishing with four units of time at the end of the cycle.

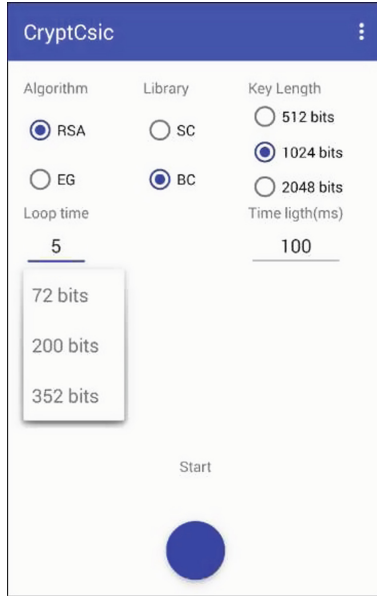


Fig. 5. Smartphone interface with some options selected

4 Experimental Results

This section presents some experimental results using SCAP Framework in the developed app.

For example, when a capture of 6 decryption iterations is released, the graph shown in Fig. 6 is obtained. In this first capture the used trigger has been a level detector (see Fig. 7). When the flash is triggered, an increase in the current supplied by the battery occurs. If the deciphering is repeated several times it is possible to see that the trace has repetitions.

In successive captures, we try to limit the moment in which the encryption/decryption process takes place, by modifying parameters of time of ignition of the flash and the length of the key. For the options of RSA, SC, 512 bits, Loop time 2, 25 ms flash time, with the same trigger, we have obtained several similar graphs (see Figs. 8 and 9). Both figures are similar, but they are not the same and it is not possible to appreciate the decryption start flashes, only the cipher start flash is noticed.

Two traces for the parameters RSA, SC, 2048 bits, Loop time 2, 50 ms flash time are shown in Figs. 10 and 11. Initially, the encryption/decryption time should be the same in each sample since the same text is always encrypted and decrypted with the same key, but there are differences between successive executions.

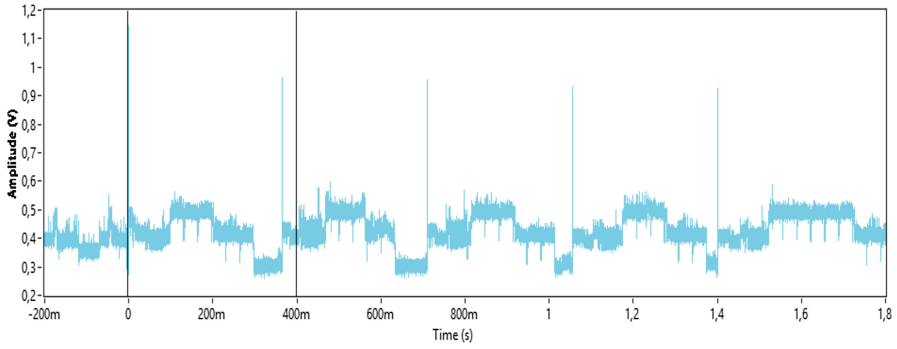


Fig. 6. Trace with 6 iterations

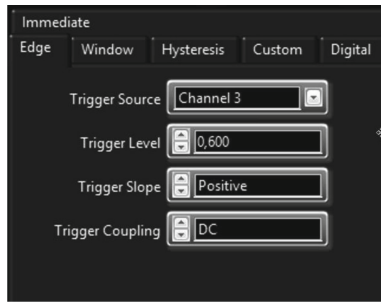


Fig. 7. Example of trigger selected

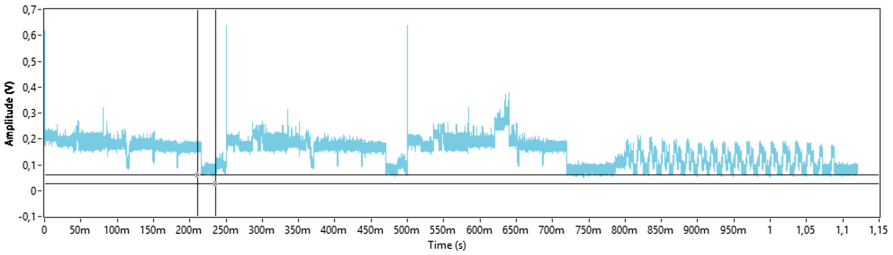


Fig. 8. Example of a trace: RSA, SC, 512 bits, loop time 2, 25 ms flash time

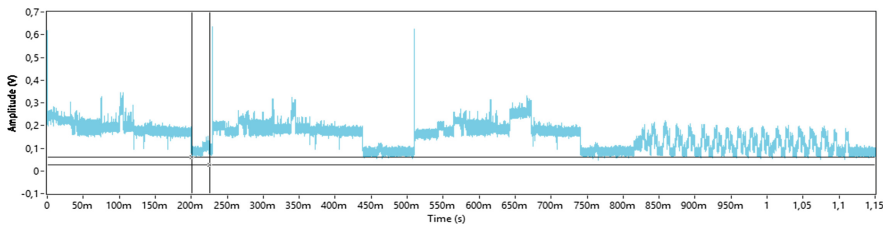


Fig. 9. A different example of trace: RSA, SC, 512 bits, loop time 2, 25 ms flash time

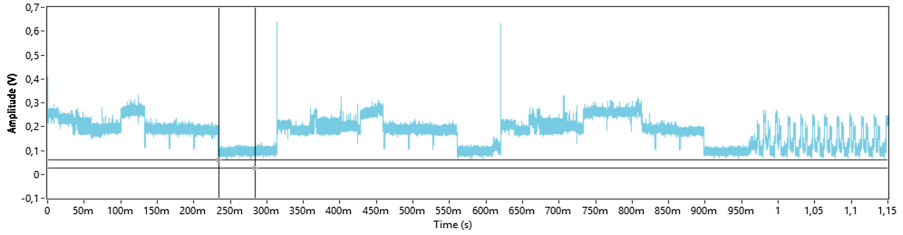


Fig. 10. Example of a trace: RSA, SC, 2048 bits, loop time 2, 50 ms flash time

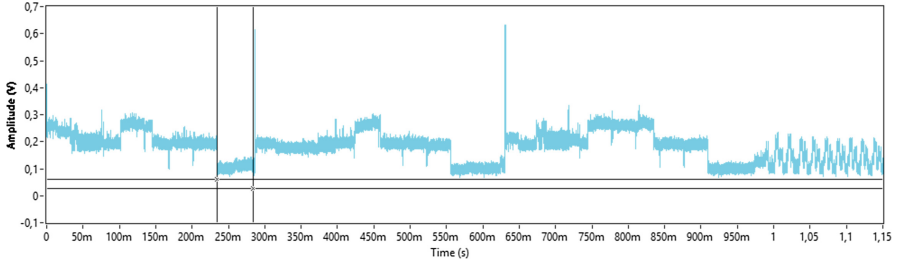


Fig. 11. A different example of trace: RSA, SC, 2048 bits, loop time 2, 50 ms flash time

5 Conclusions and Future Work

In this work we have developed a framework to capture and study the traces derived from a cryptographic device when cryptographic computations are performed. The main goal is to develop a tool for performing side-channel attacks against this type of device. The characteristics of the SCAP Framework have been described and a case study with a smartphone has been presented.

Some of the conclusions obtained from performed experiments and some of the future work to develop can be summarized as follows:

1. It is necessary to modify the software that interacts with the oscilloscope for improving the capture conditions. To do this new trigger systems, filters, etc. have to be implemented. The goal is to synchronize with much greater precision the moment when the decryption begins.
2. We have detected that the greater consumption produced in the smartphone is due to the ignition and refreshment of the screen, which masks the obtained results. Therefore, it is necessary to attenuate the backlight of the screen to obtain a less noisy trace.
3. In order to obtain a large number of traces and avoid causing vibrations in the smartphone screen that alter the position of the current probe, causing unnecessary electrical noise, it is important to automate the application with an external trigger.

4. The flash duration does not seem to be significant, so the smartphone software should be modified in order to try to limit the periods in which the flash is on. This could allow a better interpretation of the data.
5. Finally, it is necessary to modify the application of the smartphone so that it does not execute unnecessary code every time the encryption/decryption operation is launched. That is, the reading of the keys and the encryption of the text, among other things, should be optimized. This way it will be more feasible to determine the region where traces must be analyzed.

Acknowledgments. This work has been partly supported by Ministerio de Economía y Competitividad (Spain) under the projects TIN2014-55325-C2-1-R (ProCriCiS), TIN2013-46469-R (SPINY), TIN2016-79095-C2-2-R (SMOG-DEV), and by Comunidad de Madrid (Spain) under the project S2013/ICE-3095-CM (CIBERDINE), cofinanced with the European Union FEDER funds. We thank National Instruments for its support.

References

1. Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**(2), 120–126 (1978)
2. Hankerson, D., Menezes, A., Vanstone, S.: *Guide to Elliptic Curve Cryptography*. Springer, New York (2004). <https://doi.org/10.1007/b97644>
3. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) *CRYPTO 1996*. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_9
4. Wold, K., Petrovic, S.: Behavioral model of TRNG based on oscillator rings implemented in FPGA. In: *Proceedings of the 14th IEEE International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, pp. 163–166 (2011)
5. Moradi, A., Kasper, M., Paar, C.: Black-box side-channel attacks highlight the importance of countermeasures. In: Dunkelman, O. (ed.) *CT-RSA 2012*. LNCS, vol. 7178, pp. 1–18. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_1
6. De Mulder, E., Örs, S.B., Preneel, B., Verbauwhede, I.: Differential power and electromagnetic attacks on a FPGA implementation of elliptic curve cryptosystems. *Comput. Electr. Eng.* **33**(5–6), 367–382 (2007)
7. Sun, S., Yan, Z., Zambreno, J.: Experiments in attacking FPGA-based embedded systems using differential power analysis. In: *Proceedings of the IEEE International Conference on Electro/Information Technology (EIT)*, pp. 7–12 (2008)
8. Kocher, P., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. *J. Cryptogr. Eng.* **1**, 5–27 (2011)
9. Mangard, S.: Exploiting radiated emissions-EM attacks on cryptographic ICs. In: *2003 Proceedings of Austrochip*, pp. 13–16 (2003)
10. Genkin, D., Pachmanov, L., Pipman, I., Tromer, E.: Stealing keys from PCs using a radio: cheap electromagnetic attacks on windowed exponentiation. In: Güneysu, T., Handschuh, H. (eds.) *CHES 2015*. LNCS, vol. 9293, pp. 207–228. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_11. <https://eprint.iacr.org/2015/170.pdf>

11. Hutter, M., Schmidt, J.-M.: The temperature side channel and heating fault attacks. In: Francillon, A., Rohatgi, P. (eds.) CARDIS 2013. LNCS, vol. 8419, pp. 219–235. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08302-5_15. <https://eprint.iacr.org/2014/190.pdf>
12. Genkin, D., Shamir, A., Tromer, E.: RSA key extraction via low-bandwidth acoustic cryptanalysis. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 444–461. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_25. <https://www.cs.tau.ac.il/~tromer/papers/acoustic-20131218.pdf>
13. Joye, M., Tunstall, M. (eds.): Fault Analysis in Cryptography. Springer publishing, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-29656-7>
14. Anderson, R., Bond, M., Clulow, J., Skorobogatov, S.: Cryptographic processors-a survey. Proc. IEEE **94**(2), 357–369 (2006)
15. Skorobogatov, S.: Semi-invasive attacks-a new approach to hardware security analysis. Ph.D. thesis, University of Cambridge, Darwin College, UK (2005). <http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-630.pdf>
16. Kerckhoffs, A.: La cryptographie militaire. J. des Sci. Militaires **IX**, 1–2, 5–38, 161–191 (1883)
17. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_25
18. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28632-5_2
19. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks: Revealing the Secrets of Smart Cards. Advances in Information Security. Springer Science+Business Media, Heidelberg (2007). <https://doi.org/10.1007/978-0-387-38162-6>
20. Quisquater, J.-J., Samyde, D.: Electro magnetic analysis (EMA): measures and counter-measures for smart cards. In: Attali, I., Jensen, T. (eds.) E-smart 2001. LNCS, vol. 2140, pp. 200–210. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45418-7_17
21. Genkin, D., Pipman, I., Tromer, E.: Get your hands off my laptop: physical side-channel key-extraction attacks on PCs. J. Cryptogr. Eng. **5**(2), 95–112 (2015). <http://link.springer.com/content/pdf/10.1007%2Fs13389-015-0100-7.pdf>
22. Agrawal, D., Rao, J.R., Rohatgi, P.: Multi-channel attacks. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 2–16. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45238-6_2
23. Nakano, Y., Souissi, Y., Nguyen, R., Sauvage, L., Danger, J.-L., Guilley, S., Kiyomoto, S., Miyake, Y.: A pre-processing composition for secret key recovery on android smartphone. In: Naccache, D., Sauveron, D. (eds.) WISTP 2014. LNCS, vol. 8501, pp. 76–91. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43826-8_6. <https://hal.inria.fr/hal-01400921>
24. Goller, G., Sigl, G.: Side channel attacks on smartphones and embedded devices using standard radio equipment. In: Mangard, S., Poschmann, A.Y. (eds.) COSADE 2014. LNCS, vol. 9064, pp. 255–270. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21476-4_17
25. Belgarric, P., Fouque, P.A., Macario-Rat, G., Tibouchi, M.: Side-channel analysis of Weierstrass and Koblitz curve ECDSA on android smartphones. Cryptology ePrint Archive, Report 2016/231, pp. 1–26 (2016). <https://eprint.iacr.org/2016/231.pdf>

26. Belgarric, P., Fouque, P.-A., Macario-Rat, G., Tibouchi, M.: Side-channel analysis of Weierstrass and Koblitz curve ECDSA on android smartphones. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 236–252. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29485-8_14
27. Genkin, D., Pachmanov, L., Pipman, I., Tromer, E., Yarom, Y.: ECDSA key extraction from mobile devices via nonintrusive physical side channels. Cryptology ePrint Archive, Report 2016/230, pp. 1–23 (2016). <https://eprint.iacr.org/2016/230.pdf>