Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing♠

YongBin Zhou, DengGuo Feng

State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing, 100080, China

{zyb,feng}@is.iscas.ac.cn

Abstract Side-channel attacks are easy-to-implement whilst powerful attacks against cryptographic implementations, and their targets range from primitives, protocols, modules, and devices to even systems. These attacks pose a serious threat to the security of cryptographic modules. In consequence, cryptographic implementations have to be evaluated for their resistivity against such attacks and the incorporation of different countermeasures has to be considered. This paper surveys the methods and techniques employed in these attacks, the destructive effects of such attacks, the countermeasures against such attacks and evaluation of their feasibility and applicability. Finally, the necessity and feasibility of adopting this kind of physical security testing and evaluation in the development of FIPS 140-3 standard are explored. This paper is not only a survey paper, but also more a position paper.

Keywords information security, side channel attack, cryptographic module, security testing, FIPS 140

1. Introduction

Security has long been a major concern in computing and communications systems, and substantial research effort has been devoted to addressing it. Cryptographic algorithms, including symmetric ciphers, public-key ciphers, and hash functions, form a set of primitives that can be used as building blocks to construct security mechanisms that target specific objectives [115]. For example, network security protocols, such as SSH and TLS, combine these primitives to provide authentication between communicating entities, and ensure the confidentiality and integrity of communicated data. In practice, these security mechanisms only specify what functions are to be performed, irrespective of how these functions are implemented. For example, the specification of a security protocol is usually independent of whether the encryption algorithms are implemented in software running on an general processor, or using custom hardware units, and whether the memory used to store intermediate data during these computations is on the same chip as the computing unit or on a separate chip.

This kind of "separation of concerns" between security mechanisms and their implementation has enabled (and is, arguably, necessary for) rigorous theoretical analysis and design of cryptosystems and security protocols. However, in the process, various assumptions are made about the implementation of security mechanisms. For example, it is typically assumed that the implementations of cryptographic computations are ideal "black-boxes" whose internals can neither be observed nor interfered with by any malicious entity. Aided by these assumptions, the level of security is widely quantified in terms of the mathematical properties of the cryptographic

♠ The work of this paper is funded by the National Natural Science Foundation of P.R. China under the Grant No. 60503014 & No. 60273027 & No. 60373039.

algorithms and their key sizes.

In practice, however, these security mechanisms alone are far from being complete security solutions [42]. It is unrealistic to assume that attackers will attempt to directly take on the computational complexity of breaking the cryptographic primitives employed in security mechanisms. An interesting analogy can be drawn in this regard between strong cryptographic algorithms and a highly secure lock on the front door of a house [114]. Burglars attempting to break into a house will rarely try all combinations necessary to pick such a lock; they may break in through windows, break a door at its hinges, or rob owners of a key as they are trying to enter the house. Similarly, almost all known security attacks on cryptographic systems target weaknesses in the implementation and deployment of mechanisms and their cryptographic algorithms. These weaknesses can allow attackers to completely bypass, or significantly weaken, the theoretical strength of security solutions.

For a cryptographic system to remain secure it is imperative that the secret keys, that it uses to perform the required security services, are not revealed in any way. Since cryptographic algorithms themselves have been studied for a long time by a large number of experts, hackers are more likely to try to attack the hardware and system within which the cryptographic unit is housed. A new class of attacks has been developed in the last few years by Kocher [49,59]. These attacks work because there is a correlation between the physical measurements taken at different points during the computation and the internal state of the processing device, which is itself related to the secret key.

Actually, in reality, cryptographic algorithms are always implemented in software or hardware on physical devices which interact with and are influenced by their environments. These physical interactions can be instigated and monitored by adversaries, like Eve, and may result in information useful in cryptanalysis. This type of information is called side-channel information, and the attacks exploiting side-channel information are called side-channel attacks (SCA in the sequel). The underlying idea of SCA attacks is to look at the way cryptographic algorithms are implemented, rather than at the algorithm itself.

It is not difficult to see that conventional cryptanalysis treats cryptographic algorithms as purely mathematical objects, whilst side-channel cryptanalysis also takes the implementations of the algorithms into account. Hence, SCA attacks are also called implementation attacks. Even any cryptographic algorithm must be encoded in order to function properly, such encoded algorithms must not reveal the private key information used, despite the adversary's ability to observe and manipulate the running algorithm.

The first official information related to SCA attack dates back to the year 1965. P. Wright (a scientist with GCHQ at that time) reported in [113] that MI5, the British intelligence agency, was trying to break a cipher used by the Egyptian Embassy in London, but their efforts were stymied by the limits of their computational power. Wright suggested placing a microphone near the rotor-cipher machine used by the Egyptian to spy the click-sound the machine produced. By listening to the clicks of the rotors as cipher clerks reset them each morning, MI5 successfully deduced the core position of 2 or 3 of the machine's rotors. This additional information reduced the computation effort needed to break the cipher, and MI5 could spy on the embassy's communication for years.

On the other hand, the original seminal works, as well as many subsequent pioneering ideas, on SCA attacks in public cryptography research community are all due to Paul Kocher [49,59,64].

The main principles of SCA attacks are very easy to catch by. SCA attacks work because there is a correlation between the physical measurements taken during computations (e.g., power consumption, computing time, EMF radiation, etc.) and the internal state of the processing device, which is itself related to the secret key. It is the correlation between the side channel information and the operation related to the secret key that the SCA attack tries to find.

SCA attacks have been proven to be several orders of magnitude more effective than the conventional mathematical analysis based attacks and are much more practical to mount. In the area of protocol design or even software construction, one can apply a range of formal techniques to model the device in question, to model the range of adversarial actions, and then to reason about the correctness properties the device is supposed to provide nonetheless. One can thus obtain at least some assurance that, within the abstraction of the model, the device may resist adversarial attacks.

However, when we move from an abstraction notion of security to its instantiation as a real process in the physical world, things become harder. All the real-world nuances that the abstraction hid become significant. What is the boundary of this cryptographic device, in the real world? What are the outputs that an adversary may observe, and the inputs an adversary may manipulate in order to act on the device? These answers are hard to articulate, but designing an architecture to defend against arbitrary attacks requires necessarily an attempt to articulate them.

Moreover, the physical action of computation can often result in physical effects an adversary can observe; these observations can sometimes betray sensitive internal data the cryptographic module architecture was supposed to protect. This style attack of is also called side-channel analysis, since the module or device leaks information via channels other than its main intended interfaces.

By physically attacking a cryptographic device, the adversary hopes to subvert its security correctness properties somehow, usually by extracting some secret the device was not supposed to reveal. At first glance, the natural way to achieve this goal is the direct approach: somehow bypass the cryptographic modules' protections and read the data. To be fortunate, in design practice, this direct attack can be easily thwarted by so called tamper-resistant techniques. Even though this direct approach can often prove rather successful, a rather sophisticated family of indirect approaches has emerged, where the adversary instead tries to induce an error into the modules' operation via some physical failure; if the module continues to operate despite the error, it may end up revealing enough information for the adversary to reconstruct the secret. Researchers at Bellcore originally described this attack, in a theoretical context of inducing errors in cryptographic hardware that carried out the CRT implementation of RSA [90]. This result generated a flurry of follow-on results, some of which became known as differential fault analysis. These theoretical attacks eventually became practical and demonstrable, and eventually earned the name ***Bellcore attacks*** after the authors of their original paper [90].

One of the most popular jargons of system security today may be the Trusted Platform Module (TPM in the sequel). TPM usually takes the form of a cryptographically secure module and is the core of the trusted computing platform [131,169]. A key component of such cryptographic modules is that they keep and use secrets, despites attempts by an adversary — perhaps with direct physical access — to extract them.

Single-chip devices — particularly smart cards — have received much attention in the attacker community, perhaps due to the ubiquity of smart cards in low-end commerce applications

(providing motivation), and the low cost (making experiment and destructive analysis feasible for a larger population). Anderson and Kuhn's work [2,3,7] provides an enlightening (and entertaining) survey of the various techniques they found effective in practice.

Recently, two advents related to SCA research in Europe should better catch the eyes of the cryptography community worldwide, especially those who are interested in the research of SCA attacks: SCARD (Side Channel Analysis Resistant Design Flow) project [120] and ECRYPT (European Network of Excellence for Cryptology) project [121]. Both of these two projects are international joint project plans among European research members from both cryptography research institutes and relevant industries.

In SCARD, it is proposed to enhance the typical micro-chip design flow — from high level system description over register transfer layer description down to gate level net lists, and finally placement and routing of the micro-chip — in order to provide means for designing side-channel analysis resistant circuits and systems. Moreover, it is intended to study the whole phenomenon of side-channel analysis in a consistent manner, and also to provide appropriate analysis tools and to design tools for the designer of secure systems. In fact, these additional ingredients of the traditional design flow of microchips are considered to be necessary in order to enable the design of the next generation of secure and dependable devices. ECRYPT is a 4-year network of excellence funded within the Information Societies Technology Programme of the European Commission. It falls under the action line towards a global dependability and security framework and its objective is to intensify the collaboration of European researchers in information security, and more in particular in cryptology and digital watermarking. In order to reach this goal, 32 leading players integrate their research capabilities within five virtual labs focused on different core research areas, with one being secure and efficient implementations (VAMPIRE). One of the four Working Groups of VAMPIRE is the research group on SCA analysis.

From these two advents alone, it is roughly estimated that the Europe, in our own opinion, is likely one step further over the other continents in the internationally collaborative research on SCA attacks.

It is an interesting story that SCA attacks evaluation was already explicitly suggested many years ago to be encompassed in cryptographic algorithm evaluation in many international standards bodies, such as 3GPP security architecture [8]. However, due to lack of testable methods and practical tools, this insightful suggestion virtually is like vacant shapes in sight. So it is very easy to understand that the final evaluation report of these standard bodies draw the conclusion at that time that "in the design process it was concluded not to be feasible to design a general algorithm framework that by itself would not be vulnerable to side channel attacks" [119].

Recently, Tiri and Verbauwhede presented a digital VLSI design flow to create secure, side-channel attack resistant integrated circuits (IC in the sequel) [66]. Even though this is the first significant attempt in the secure design of IC, they only considered the power analysis attack in the comprehensive top-down automated synchronous VLSI design flow that pursues a constant power dissipation. Kocher et al. [64] proposed the point of view that security should be treated as a intrinsic dimension in embedded system design. Ravi et al. [63] discussed the general tamper resistant mechanisms for secure embedded systems. They developed a preliminary systematic security embedded system design approach. In their case study, the concept of trusted code base was introduced, which resembles the trusted computing base in the context of secure operating system.

The threat of SCA attacks also caught the attention from NoC research community [11]. Gebotys et al. presented a framework for security of NoCs by providing network level symmetric key cryptography for key distribution and at the core level by illustrating modification of software with extremely low overheads for added security against power attacks [11].

Clearly, a cryptographic algorithm which is strong with respect to conventional cryptanalytic attacks is useless if it cannot be implemented securely on a broad range of platforms. Already during the AES and NESSIE processes, the cryptographic community has come to this conclusion.

Some motivations of this paper are as follows: to understand the history of SCA attacks; to recognize the serious threats of SCA attacks; to acknowledge the various countermeasures against SCA attacks; to evaluate the impacts of SCA attacks on the security testing of cryptographic modules; to identify the possible research trends in this area and so on.

The remainder of this paper is organized as follows. In section 2, we present the models for side channel attacks. FIPS 140 standard is briefly recalled in section 3, and then some problems about the current version of this standard are identified. In Section 4, classification of SCA attacks is discussed. In section 5, we present concrete side-channels discovered so far and the relevant countermeasures. In section 6, we give out some thoughts about the possible impacts of SCA attacks on cryptographic module security testing. Concluding remarks are given in Section 7.

2. Models of Side Channel Attacks

A cryptographic primitive can be considered from at least two points of views: on the one hand, it can be viewed as an abstract mathematical object (a transformation, possibly parameterized by a key, turning some input into some output); on the other hand, this primitive will in fine have to be implemented in a program that will run on a given processor, in a given environment, and will therefore present specific characteristics. The first point of view is that of "classical" cryptanalysis; the second one is that of side-channel cryptanalysis. Side-channel cryptanalysis takes advantage of implementation-specific characteristics to recover the secret parameters involved in the computation. It is therefore much less general — since it is specific to a given implementation — but often much more powerful than classical cryptanalysis, and is considered very seriously by cryptographic devices' implementors.

In traditional cryptanalysis, when assessing the security of a cryptographic protocol, one usually assumes that the adversary has a complete description of the protocol, is in possession of all public keys, and is only lacking knowledge of the secret keys. In addition, the adversary may have intercepted some data exchanged between the legitimate participants, and may even have some control over the nature of this data (e.g., by selecting the messages in a chosen-message attack on a signature scheme, or by selecting the ciphertext in a chosen-ciphertext attack on a public-key encryption scheme). The adversary then attempts to compromise the protocol goals by either solving an underlying problem assumed to be intractable, or by exploiting some design flaw in the protocol.

In this process, mathematical abstraction can be a very useful tool in the study of cryptographic primitives. Cryptographers often evaluate the security of ciphers by considering them as mathematical functions used in the scenario similar to the one described in Figure 1.

Traditionally, secure cryptographic algorithms provide security against an adversary who has only ***black-box*** access to the secret information of honest parties. However, such models are not always adequate. In particular, the security of these algorithms may completely break under

(feasible) attacks that try to tamper with the secret key.

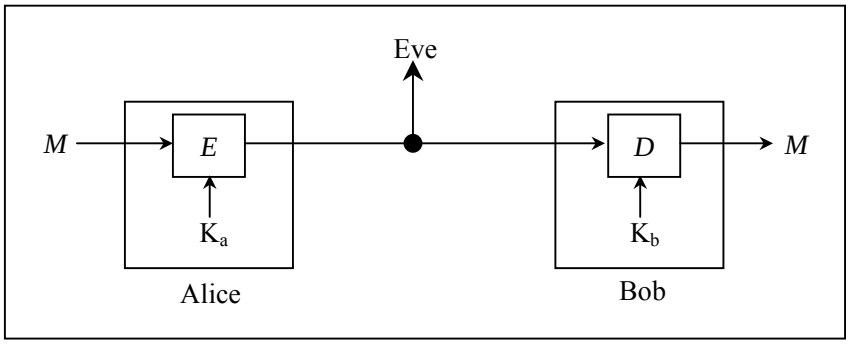

Figure 1: The traditional cryptographic model

The attacks considered in this traditional security model exploit the mathematical specification of the protocol. In recent years, researchers have become increasingly aware of the possibility of attacks that exploit specific properties of the implementation and operating environment. Such SCA attacks utilize information leaked during the protocol's execution and are not considered in traditional security models. For example, the adversary may be able to monitor the power consumed or the electromagnetic radiation emitted by a smart card while it performs private-key operations such as decryption and signature generation. The adversary may also be able to measure the time it takes to perform a cryptographic operation, or analyze how a cryptographic device behaves when certain errors are encountered. Side-channel information may be easy to gather in practice, and therefore it is essential that the threat of SCA attacks be quantified when assessing the overall security of a system, see the scenario illustrated in Figure 2.

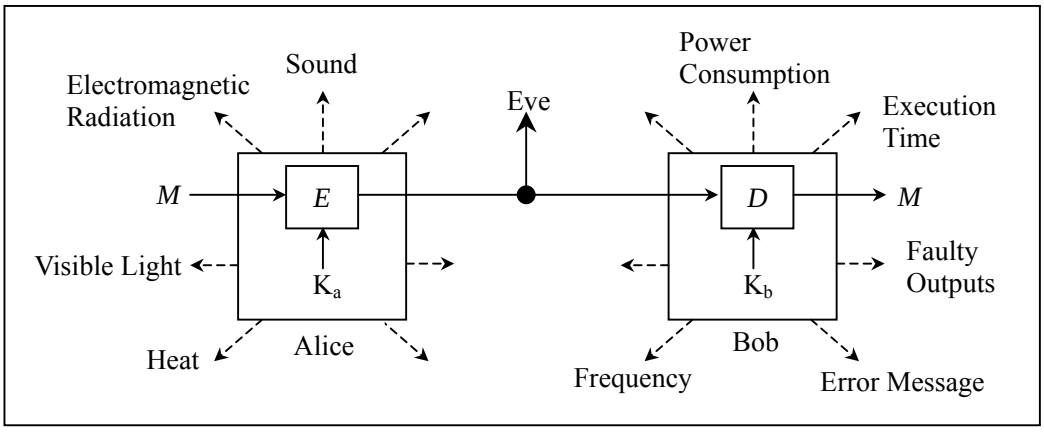

Figure 2: The cryptographic model including side-channel

Side Channels are defined to be unintended output channels from a system. Paul Kocher in 1996 published the seminal paper "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems" showing that non-constant running time of ciphers can leak information about the key. When implementations take advantage of optimizations, the problem may become more pronounced.

It should be emphasized that a particular side-channel attack may not be a realistic threat in some environments. For example, attacks that measure power consumption of a cryptographic device can be considered very plausible if the device is a smart card that draws power from an external, untrusted source. On the other hand, if the device is a workstation located in a secure

office, then power consumption attacks are not a significant threat.

3. FIPS 140 Standard and Some Related Problems

FIPS 140 is the standard to be used by (US) Federal organizations when specifying cryptographic-based security systems to provide protection for sensitive or valuable data (maintaining the confidentiality and integrity of information). The FIPS 140 standard specifies the security requirements to be satisfied by a cryptographic module in four increasing, qualitative levels of security (Level 1 to 4, from low to high) as summarized in the following:

- ***Security Level 1*** provides the lowest level of security. It specifies basic security requirements for a cryptographic module. (For software implementation only).
- ***Security Level 2*** improves the physical security of a Security Level 1 cryptographic module by adding the requirement for tamper evident coatings or seals, or for pick-resistant locks.
- ***Security Level 3*** requires enhanced physical security, attempting to prevent the intruder from gaining access to critical security parameters held within the module.
- ***Security Level 4*** provides the highest level of security. Level 4 physical security provides an envelope of protection around the cryptographic module to detect a penetration of the device from any direction.

These levels are intended to cover the wide range of potential applications and environments in which cryptographic modules may be employed. The security requirements cover eleven areas related to the secure design and implementation of the cryptographic module. These areas include the following: cryptographic module specification; cryptographic module ports and interfaces; roles, services, and authentication; finite state model; physical security; operational environment; cryptographic key management; electromagnetic interference/electromagnetic compatibility (EMI/ EMC); self tests; design assurance and mitigation of other attacks.

The FIPS 140 standard is reexamined or reaffirmed every five years. FIPS 140-1 standard specifies the security requirements to be satisfied by a cryptographic module used within a security system protecting unclassified information within computer and telecommunications systems (including voice systems). FIPS 140-2 superseded FIPS 140-1 in 2001 with technical modifications to address technological advances that had occurred since FIPS 140-1 had been issued.

Actually, there are some problems with the current version of FIPS 140-2. First of all, this version of standard is mainly focused on hardware modules, and is not well adapted to software modules. It is expected this status may change in the coming version of FIPS 104-3. Secondly, this version of standard covers somewhat too narrow scopes of the system to be tested. Better alignment with the Common Criteria is required, and the security vulnerabilities of functional protocols need to be addressed better. Finally, the functional requirements of this version of standard are already out-of-date. The requirements specified in FIPS 140-2 have lagged behind the actual needs of the information security both in theory and practice.

Meanwhile, the complex process of FIPS 140 validation shows that excellence in creating solid cryptographic algorithms and modules is difficult to achieve. Of the eleven areas, the following four areas are likely of greatest difficulty: physical security, self-tests, random number generation and key management.

Specifically, as far as SCA attack is concerned, FIPS 140-1 did not explicitly mention the

security of cryptographic modules against side channel attacks, while FIPS 140-2 only deal briefly with the specification of mitigation of attacks for which no testable requirements are currently available.

4. Classifications of Side Channel Attacks

Side channel attacks are usually classified in literatures along the following three orthogonal axes:

- Classifications depending the control over the computation process;
- Classifications depending on the way of accessing the module;
- Classifications depending on the method used in the analysis process.

4.1 Controls over the Computation Process

Depending on the control over the computation process by attackers, SCA attacks can be broadly divided into two main categories: ***passive attacks*** and ***active attacks***. We refer passive attacks to those that do not noticeably interfere with the operation of the target system; the attacker gains some information about the target system's operation, but the target system behaves exactly as if no attack occurs. In active attack, on the other hand, the adversary exerts some influence on the behavior of the target system. While the actively attacked system may or may not be able to detect such influence, an outsider observer would notice a difference in the operation of the system. It is important to note that the distinction between active attacks and passive attacks has more to do with the intrinsic nature of the attack than the intrusiveness of ant physical implementation of the attack.

4.2 Ways of Accessing the Module

When analyzing the security of a cryptographic hardware module, it can useful to perform a systematic review of the attack surface — the set of physical, electrical and logical interfaces that are exposed to a potential opponent. According to this observation, Anderson et al. [111] divided the attacks into the following classes: ***invasive attacks***, ***semi-invasive attacks*** and ***non-invasive attacks.***

4.2.1 Invasive Attacks

An ***Invasive attack*** involves depackaging to get direct access to the internal components of cryptographic modules or devices. A typical example of this is that the attackers may open a hole in the passivation layer of a cryptographic module and place a probing needle on a data bus to see the data transfer.

Tamper resistant or responsive mechanisms are usually implemented in hardware to effectively counter invasive attacks. For example, some cryptographic modules of higher security level will zeroize all their memories when tampering are detected [116].

4.2.2 Semi-invasive Attacks

The concept of ***semi-invasive attack*** is first developed by Skorobogatov and Anderson [95]. This kind of attack involves access to the device, but without damaging the passivation layer or

making electrical contact other than with the authorized surface. For example, in a fault-induced attack, the attacker may use a laser beam to ionize a device to change some of its memories and thus change the output of this device.

4.2.3 Non-invasive Attacks

A ***non-invasive attack*** involves close observation or manipulation of the device's operation. This attack only exploits externally available information that is often unintentionally leaked. A typical example of such an attack is timing analysis: measuring the time consumed by a device to execute an operation and correlating this with the computation performed by the device in order to deduce the value of the secret keys.

One important characteristic of non-invasive attack is that this attack is completely undetectable. For example, there is no way for a smart card to figure out that its running time is currently being measured. On the other hand, compared with invasive attacks that require individual processing of each attacked device, non-invasive attacks are usually of low-cost to deploy on a large scale from an economical point of view. In this sense, non-invasive attacks constitute therefore a bigger menace for the smart card industry.

4.3 Methods Used in the Analysis Process

Depending on the methods used in the process of analyzing the sampled data, SCA attacks can be divided simple side channel attack (***SSCA*** in the sequel) and differential side channel attack (***DSCA*** in the sequel).

In a SSCA, the attack exploits the side-channel output mainly depending on the performed operations. Typically, a single trace is used in an SSCA analysis, and therefore the secret key can be directly read from the side-channel trace. Obviously, the side-channel information related to the attacked instructions (the signal) needs to be larger than the side-channel information related to the unrelated instructions (the noise) [154]. What SSCA exploits is the relationship between the executed instructions and the side-channel output.

On the other hand, when SSCA is not feasible due too much noise in the measurements, DSCA using statistical methods is tried. What DSCA exploits is the correlation between the processed data and the side-channel output.In DSCA, the attack exploits the side-channel output mainly depending on the performed data. Typically, many traces are used in a DSCA analysis, and then statistical methods are used to deduce the possible secret keys. With regard to this, one can claim that DSCA is more powerful than SSCA.

Differential side-channel attacks exploit the correlation between the data and the instantaneous side-channel leakage of the cryptographic device. As this correlation is usually very small, statistical methods must be used to exploit it efficiently. In a differential side-channel attack, an attacker uses a hypothetical model of the device under attack. The quality of this model depends on the capabilities of the attacker.

The hypothetical model is used to predict the side-channel output of the device; it may output several values. These could be either values describing one type of information leakage for several time slots, or it could be values predicting the leakage of different side-channels. In case only one single output-value is used for an attack, then the attack is called ***first-order attack***. If two or more output values for the same side-channel are used in an attack, then the attack is called ***second-order attack*** and ***higher-order attack***, respectively.

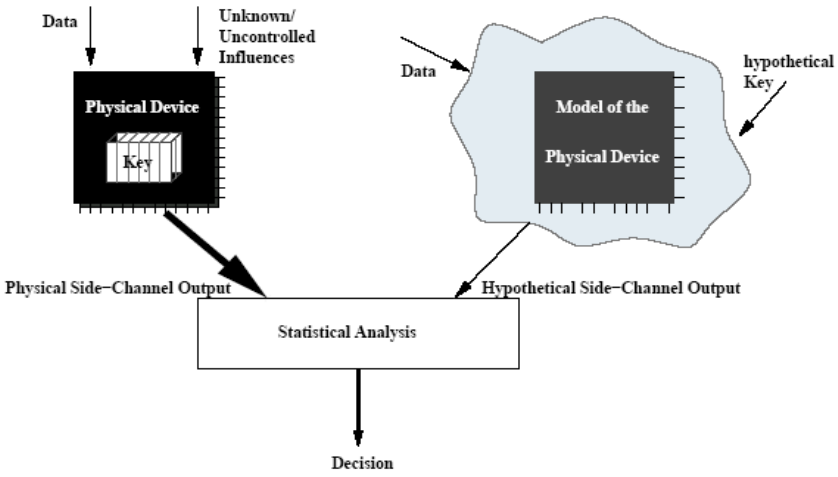

Figure 3: The general idea behind DSCA [132]

Remarks ***The above three axes sometimes are well orthogonal: an invasive attack may completely avoid disturbing the device's behavior, and a passive attack may require a preliminary depackaging for the required information to be observable, an active and invasive attack may also belong to the DSCA.***

5. Known Side Channel Attacks and Concrete Countermeasures

This section is the main part of the paper. In this section, we will review the known methods and techniques employed in SCA attacks, the destructive effects of such attacks, the countermeasures proposed so far against such attacks and evaluation of their feasibility and applicability.

So far, SCA attacks have been successfully used to break the hardware or software implementations of many cryptosystems including block ciphers(such as DES ,AES, Camellia, IDEA, Misty1, etc.), stream ciphers(such as RC4, RC6 ,A5/1, SOBER-t32, etc.), public key ciphers(such as RSA-type ciphers, ElGamal-type ciphers, ECC, XTR, etc.), to break the implementations of signature schemes, to break the message authentication code schemes, to break the implementation of cryptographic protocols, to break the implementation of cryptosystems, and even to break the networking systems.

As many security experts have pointed out, security does not equal to cryptography and good cryptographic algorithms do not automatically guarantee the security of application systems. Every component is secure does not necessarily mean that the whole system is secure. For complex systems, security should be studied under various attacks from ***various angles*** very carefully. Without doubt, SCA attack is definitely such kind of useful angel to be explored more throughly.

5.1 Known Side Channel Attacks

SCA attacks against cryptographic modules exploit characteristic information extracted from the implementation of the cryptographic primitives and protocols. This characteristic information can be extracted from timing, power consumption or electromagnetic radiation features. Other

forms of side-channel information can be a result of hardware or software faults, computational errors, and changes in frequency or temperature. SCA attacks make use of the characteristics of the hardware and software elements as well as the implementation structure of the cryptographic primitive. Therefore, in contrast to analyzing the mathematical structure and properties of the cryptographic primitives only, side-channel analysis also includes the implementation.

All these facts sum up to one fact that the concrete implementation is very critical to security and a tiny difference in implementations could make a big difference in security. Hence engineers who implement the security schemes should be very carefully in following every step of the schemes. Moreover, attackers will more likely choose the weakest link in the security chain. When peer reviewed cryptographic algorithms and protocols are used, cryptanalysis will almost certainly not be the weakest link. Systems designers must strive to be aware of unintentional “back doors” which are not secure against attacks.

Until today, at least more than ten kinds of important side channels have been explored. We will discuss them one by one in this section.

5.1.1 Timing Attack

Implementations of cryptographic algorithms often perform computations in non-constant time, due to performance optimizations. If such operations involve secret parameters, these timing variations can leak some information and, provided enough knowledge of the implementation is at hand, a careful statistical analysis could even lead to the total recovery of these secret parameters. This idea was introduced by Kocher [49], and was developed by Dhem et al. [59] in which a practical timing attack against an actual smart card implementation of the RSA was conducted.

A timing attack is, essentially, a way of obtaining some user's private information by carefully measuring the time it takes the user to carry out cryptographic operations. The principle of this attack is very simple: to exploit the timing variance in the operation.

The basic assumptions of timing analysis are:

①. The run time of a cryptographic operation depends to some extent on the key. With present hardware this is likely to be the case, but note that there are various efficient hardware based proposals to make the timing attack less feasible through ‘noise injection’. Software approaches to make the timing attack infeasible are based on the idea that the computations in two branches of a conditional should take the same amount of time (‘branch equalisation’).

②. A sufficiently large number of encryptions can be carried out, during which time the key does not change. A challenge response protocol is ideal for timing attacks.

③. Time can be measured with known error. The smaller the error, the fewer time measurements are required.

Timing attacks were introduced in 1996 by Kocher [49], where RSA modular exponentiation was being attacked. Schindler presented timing attacks on implementation of RSA exponentation that employ the Chinese Remainder Theorem (CRT in the sequel) [155]. Experimental results for an RSA implementation on a smart card were reported by Dhem et al. [55]. Timing attacks on DES that recover the Hamming weight of the secret key were described by Hevia et al. [156].

OpenSSL is a well-known free (open source) crypto library which is often used on Apache Web Servers to provide SSL functions. Brumley and Boneh [54] demonstrated that timing attacks can reveal RSA private keys from an OpenSSL-based web server over a local network. They showed that a modified version of Kocher's attack can be carried out remotely against servers

running OpenSSL. By making ~1/3 million queries (~2 hours), factors of a 1024 bit modulus can be found. Canvel et al. [157] devised timing attacks on the CBC-mode encryption schemes used in SSL and TLS; their attacks can decrypt commonly used ciphertext such as the encryption of a password.

At the 10th Usenix Security Symposium, Song et al. [50] presented timing analysis of keystrokes and timing Attacks on SSH protocol. They applied traffic-analysis techniques to interactive SSH connections in order to infer information about the encrypted connection contents. They concluded that the keystroke timing data observable from SSH implementations reveals a dangerously significant amount of information about user terminal sessions — enough to locate typed passwords in the session data stream and reduce the computational work involved in guessing those passwords by a factor of 50.

Cathalo et al. [51] proposed a timing attack on the GPS identification scheme of NESSIE project in Europe [158]. They showed that only 800 timing measurements allow the attacker to find the private key in a few seconds on a PC with a success probability of 80%. Interestingly, their attacking methods resist some classical countermeasures and work whether the Chinese Remainder technique is used or not.

By observing the timing of the reject signs from the decryption oracle, Sakurai et al. [52] presented a timing attack against the EPOC-2 public-key cryptosystem that was proved to be IND-CCA2 secure under the factoring assumption in the random oracle model. More interestingly, EPOC-2 was already written into a standard specification P1363 of IEEE, and has been a candidate of the public-key cryptosystem in several international standards (or portfolio) on cryptography, e.g. NESSIE, CRYPTREC, ISO, etc.

Recently, Levine et al. [53] presented a timing attack against low latency MIX-based systems that are communication proxies that attempt to hide the correspondence between its incoming and outgoing messages. A novel technique, defensive dropping, was also proposed by them to thwart timing attacks. Some one argues that mounting this attack over the network is unlikely to be successful because time measurement is too inaccurate. However, we believe that in a distributed system with real-time properties, timing attacks on the security protocols of such system may become a real threat.

One simple defense approach is to make the operational parameters independent of the input data. The feasibility of this approach depends on the operation. For example, in RSA, one can use random data to conduct a blinding transformation (a.k.a. ***noise injection***) on the parameters before the operation, and then a reverse unblinding transformation afterwards. However, carrying out this approach on a trusted computing platform that does not have a good source of randomness — or a good way to obtain a seed and store a context of pseudorandomness — can be tricky. Actually, since Kocher's original paper (1996) users of RSA have been strongly recommended to use blinding. Some software implementations (such as Netscape's cryptography code) did blind, however many implementations (OpenSSL, GnuTLS, GPG, and more) did not (this has been fixed). By the way, blinding adds about 2%-10% overhead. And another countermeasure to attacks of this type is to eliminate branch processing in the implementing algorithm so that encryption times are equivalent (a.k.a. ***branch equalisation***).

Actually, countermeasures for timing attacks must be modelled more rigorously so that we can study how effective the proposed measures are. Two common countermeasures that are currently in use (i.e. noise injection and branch equalization) appear to be fundamentally different

in the sense that noise injection weakens the power of the timing attack but it does not defeat it, whereas branch equalisation does defeat the attack but at significant cost.

Another easier defense approach — and one that newer-generation modular exponentiation and RSA engines started to incorporate — is to design the hardware to take constant time for each operation, no matter what the data was. When Paul Kocher first published his timing attacks in 1995, at least one old-timer claimed that a few older commercial accelerators also took constant time, indicating that some people in the commercial world must have already known about the attack.

It it worth noting that even the timing attack exploits the timing variation in each operation of the algorithm, the individual timing of each operation can not be measured in practice. Only the total executing time of all the operations of the algorithm can be measured, and then statistical methods are being applied to deduce (part of) the secret key.

5.1.2 Fault Attack

Most of the devices that perform various cryptographic operations are usually assumed to operate reliably when we use them, so we might not think to question if the security of such operations depend on the reliability of these devices that implement them. In spite of this assumption, hardware faults and errors occurring during the operation of a cryptographic module in fact have been demonstrated to seriously affect the security. These faulty behaviors or outputs may also become important side channels, and will even greatly increase a cipher's vulnerability to cryptanalysis sometimes. Fault attacks present practical and effective attacking against the cryptographic hardware devices such as smart cards. Therefore, we mainly focus on the fault attacks on hardware devices here.

Fault attacks on cryptographic algorithms have been studied since 1996 [90] and since then, nearly all the cryptographic algorithms have been broken by using such kinds of attacks. Fault attacks offer the attacker plenty of possibilities to attack a cryptosystem. The ways to exploit a faulty result are very different from one algorithm to another. The feasibility of a fault attack (or at least its efficiency) depends on the exact capabilities of the adversary and the type of faults she can induce. Generally, a ***fault model*** should at least specify the following aspects:

- The precision an attacker can reach in choosing the time and location on which the fault occurs during the execution of a cryptographic module.
- The length of the data affected by a fault; for example, only one bit, or one byte.
- The persistence of the fault; whether the fault is transient or permanent.
- The type of the fault; such as flip one bit; flip one bit, but only in one direction (e.g. from 1 to 0); byte changed to a random (unknown) value; and so on.

There are two major kinds of fault side channels. The first ones are channels which are induced by computational faults occurring during cryptographic computation in an attacked module. These faults can be either random or intentional, caused, for instance, by a precise voltage manipulation [2,3]. Having the ability to introduce computational faults, this kind of attack can be used on almost every kind of cryptographic mechanism and it is regarded as one of the the most effective side channel attacks at all. The second kinds of fault side channels are those which are induced by sending an intentionally corrupted input data to the attacked module. For the module, this means a non-standard situation which must be handled in a special way. Usually the module has to use an error message to inform the user (the module can hardly know whether this is an

ordinary user or an attacker) that the computation has been stopped due to some reasons.

Generally speaking, a successful fault attack on cryptographic modules or devices requires two steps: the fault injection and the fault exploitation steps. These two steps are illustrated in Figure 4. The first step consists in injecting a fault at the appropriate time during the process. Fault injection is very dependent on the devices' hardware. Faults can be induced in a smart card by acting on its environment and putting it in abnormal conditions. Some of them are abnormally and abruptly low or high voltage, clock, temperature, radiations, light, and so on. The issue of fault induction techniques was addressed in many literatures, and we refer you to [2,3]. The second step consists in exploiting the erroneous result or unexpected behavior. Fault exploitation depends on the software design and implementation. In case of an algorithm it will also depends on its specification since the fault exploitation will be combined with cryptanalysis most of the time. Depending on the type of analysis performed, the fault injection will have to be done at a precise instant or roughly in a given period of time.

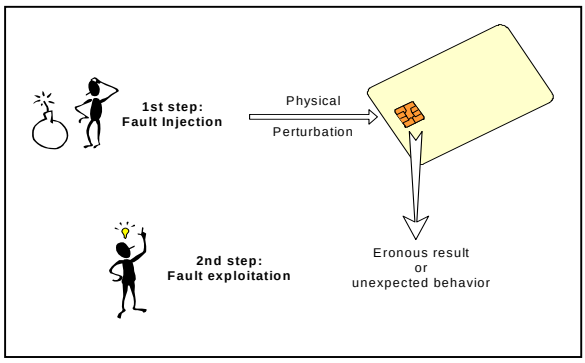

Figure 4: Fault attacks against a smart card

Fault analysis attacks were first considered in 1997 by Boneh et al.[90,110], who described such attacks on the RSA signature scheme and the Fiat-Shamir and Schnorr identification protocols. Bao et al.[104] presented fault analysis attacks on the ElGamal, Schnorr and DSA signature schemes. Fault analysis attacks on elliptic curve public-key encryption schemes were presented by Biehl et al. [93]. Their attacks succeed if an error during the decryption process produces a point that is not on the valid elliptic curve. The attacks can be prevented by ensuring that points that are the result of a cryptographic calculation indeed lie on the correct elliptic curve. Biham and Shamir [92] presented fault analysis attacks on the DES symmetric-key encryption scheme. Anderson and Kuhn [3] discussed some realistic ways of inducing transient faults, which they call glitches.

Skorobogatov and Anderson [95] proposed a powerful yet surprisingly practical optical fault attack. They demonstrated that inexpensive equipment can be used to induce faults in a smart card by illuminating specific transistors; they also proposed countermeasures to these optical fault induction attacks. This attack can again convince the reader that fault injection is definitely a problem worth considering in the design and testing of a secure system or device.

Differential fault analysis (DFA in the sequel) [92] usually causes some sorts of physically erroneous operation to occur in a cryptographic device and then measures resulting phenomena. They mainly consist in analyzing an algorithm result (ciphertext) under regular condition and under abnormal condition for the same input (plaintext). The abnormal condition is usually obtained by fault injection during the process (transient fault) or before the process (permanent fault). DFA has been widely studied from a theoretical point of view, and seemed to be applicable to almost all symmetric cryptosystems.

If we suppose that an attacker cannot induce the same fault twice, one of the best countermeasures to protect the symmetric algorithms such DES and AES is to compute the whole or a part of the rounds twice (including key scheduling). Certainly, this will degrade the whole performance. In case of the public key algorithms, one defense approach is to verify the signature (the output of private key operations) by using the public key before sending the signature out. More interstingly, Yen et al. [96] showed that checking the correctness of the computed result before giving it to others may not be enough to prevent a hardware fault-based cryptanalysis.

Another countermeasure suggested to protect public key algorithms from some specific fault attacks is to check the integrity of the secret key at the end of signature computation. Other general tricks irrespective of concrete algorithms were also proposed, including checksums, execution randomization, ratification counters and baits, repeated refreshments [102].

To summarize, fault attacks are real and big threats for any secure token (whatever the form factor) and must be taken into consideration at all steps of the product design and specification. Countermeasure and protection against fault attacks can be designed in both hardware and software. Devising and analyzing fault attacks are necessary as they permit us to estimate the strength of the countermeasures to be deployed.

5.1.3 Power Analysis Attack

In addition to its running time and its faulty behaviour, the power consumption of a cryptographic device may provide much information about the operations that take place and the involved parameters. This is the very idea of power analysis attack. Certainly, power analysis attack is applicable only to hardware implementation of the cryptosystems. Power analysis attack is particularly effective and proven successful in attacking smart cards or other dedicated embedded systems storing the secret key.

Of all types of SCA attacks known today, the number of literatures on power analysis attacks and the relevant countermeasures is the biggest. Roughly calculating, there are at least more than 200 papers published currently in this area. Power analysis attack is actually the current research focus of side-channel attacks.

Power analysis attacks have been demonstrated to be very powerful attacks for most straightforward implementations of symmetric and public key ciphers [30,31,32,33,34]. For simplicity, we use Elliptic Curve Cryptosystems (ECC) to illustrate the power analysis attacks in this section. Yet, many of the relevant attacking methods and various countermeasures are applicable also to other cryptosystems.

Basically, power analysis attack can be divided into Simple and Differential Power Analysis (referred to as SPA and DPA, respectively). In SPA attacks, the aim is essentially to guess from the power trace which particular instruction is being executed at a certain time and what values the input and output have. Therefore, the adversary needs an exact knowledge of the implementation to mount such an attack. On the other hand, DPA attack does not need the knowledge of the implementation details and alternatively exploiting statistical methods in the analysis process. DPA is one of the most powerful SCA attacks, yet it can be mounted using very little resources.

More advanced differential power analysis looks at subtle statistical correlations between the secret bits and power consumption. DPA is a strong attack, but it only works in certain cases (e.g. smartcards).In its classic instantiation, the adversary collects a large set $\{T_i, C_i\}$ of trace-ciphertext pairs. The adversary also picks a selection function D that takes a ciphertext and a guess of part

of the key and outputs one bit. The idea is that if the guess is right, this bit reflects something that actually shows up in the computation, but if the guess is wrong, then D will be random across the ciphertexts.

The adversary then makes a guess K_g and uses this guess and the selection function D to partition the set of traces into two sets: the one for which $D(C_i, K_g) = 0$ and the other one for which $D(C_i, K_g) = 1$. He averages the traces in each set, and then looks at the difference between these average traces. If K_g was wrong, these two sets are uncorrelated, and the differential trace becomes flat as the sample size increases. However, if K_g was right, the differential approaches the correlation of D and power consumption, which will be spiky.

SPA and DPA attacks were introduced in 1999 by Kocher et al. [59]. They carried out a practical power analysis attack against an DES implementation in hardware. Coron [13] was the first to apply these attacks to elliptic curve cryptographic schemes, and proposed the SPA-resistant method for point multiplication, and the DPA-resistant method of randomizing projective coordinates. Oswald [159] showed how a multiplier k can be determined using the partial information gained about ***NAF(k)*** from a power trace of an execution of the binary NAF point multiplication method. Experimental results with power analysis attacks on smart cards were reported by Akkar et al. [160] and Messerges et al. [31], while those on a DSP processor core are reported by Gebotys et al. [161].

Chari et al. [75] presented some general SPA and DPA countermeasures, and a formal methodology for evaluating their effectiveness. Proposals for hardware-based defenses against power analysis attacks include using an internal power source, randomizing the order in which instructions are executed (May et al. [162]), randomized register renaming (May et al. [83]), and using two capacitors, one of which is charged by an external power supply and the other supplies power to the device (Shamir [163]).

One effective method for guarding against SPA attacks on point multiplication is to employ elliptic curve addition formulas that can also be used for doubling. This approach was studied by Liardet et al. [15] for curves in Jacobi form, by Joye er al. [17] for curves in Hessian form, and by Brier and Joye [16] for curves in general Weierstrass form. Izu et al. [164] devised an active attack (not using power analysis) on the Brier-Joye formula that can reveal a few bits of the private key in elliptic curve schemes that use point multiplication with a fixed multiplier. Hasan [168] studied power analysis attacks on point multiplication for Koblitz curves and proposed some countermeasures which do not significantly degrade performance.

Another strategy for SPA resistance is to use point multiplication algorithms such as Coron's method [13] where the pattern of addition and double operations is independent of the multiplier. Other examples are Montgomery point multiplication (see Okeya et al.'s methods [18]), and the methods presented by Möller [22,24], Hitchcock et al. [165], and Izu and Takagi [21]. The security and efficiency of (improved versions) of the Möller [22] and Izu-Takagi [21] methods were carefully analyzed by Izu et al. [166]. Another approach taken by Trichina et al. [167] and Gebotys et al. [161] is to devise formulas for the addition and double operations that have the same pattern of field operations (addition, subtraction, multiplication and squaring).

Joye et al. [14] proposed using a randomly chosen elliptic curve isomorphic to the given one, and a randomly chosen representation for the underlying fields, as countermeasures to DPA attacks. Goubin [25] showed that even if point multiplication is protected with an SPA-resistant method (such as Coron's method [13]) and a DPA-resistant method (such as randomized projective

coordinates, randomized elliptic curve, or randomized field representation), the point multiplication may still be vulnerable to a DPA attack in situations where an attacker can select the base point (as is the case, for example, with ECIES). Goubin's observations highlight the difficulty in securing point multiplication against power analysis attacks.

The SPA simply observes several power consumptions of the device, and the DPA is additionally allowed to use a statistical tool in order to guess the secret information. An SPA-resistant scheme can be converted to be a DPA-resistant one by randomizing the parameters of the underlying system (See [13,14],for example).

There are three different types of SPA-resistant schemes, available at present , for ECC scalar multiplication: (1) indistinguishable addition formula that uses one formula for both of elliptic addition and doubling [15,16,17]; (2) addition chain that always computes elliptic addition and doubling for each bit [13,16,18,20,21]; (3) window based addition chain with fixed pattern [19,22,23,24].

Defenses against differential power analysis are difficult, since they essentially only reduce the signal the adversary is reading, rather than eliminate it. Interestingly, an efficient randomization technique, using some random variables within the point addition operation, has also been proposed as a possible countermeasure against a DPA-style attack on the window-family algorithm in [29].

5.1.4 EM Attack

As electrical devices, the components of a computer often generate electromagnetic radiation as part of their operation. An adversary that can observe these emanations and can understand their causal relationship to the underlying computation and data may be able to infer a surprising amount of information about this computation and data. This ability can be devastating, should the computer be a trusted computing platform intended to keep this information from the adversary.

Similar to the power analysis attacks, ElectroMagnetic Analysis (EMA) attacks can also be divided into two main categories: Simple ElectroMagnetic Analysis (SEMA) and Differential ElectroMagnetic Analysis (DEMA).

The potential of exploiting electromagnetic emanations has been known in military circles for a long time. For example, see the recently declassified TEMPEST document written by the National Security Agency [136] that investigates different compromising emanations including electromagnetic radiation, line conduction, and acoustic emissions. The unclassified literature on attack techniques and countermeasures is also extensive. For example, Kuhn et al. [138] discuss software-based techniques for launching and preventing attacks based on deducing the information on video screens from the electromagnetic radiations emitted. Experimental results on electromagnetic analysis attacks on cryptographic devices such as smart cards and comparisons to power analysis attacks were first presented by Quisquater et al. [137] and Gandolfi et al. [135]. The most comprehensive unclassified study on EMA attacks to date is the work of Agrawal et al. [133]. They showed that not only can EM emanations be used to attack cryptographic devices where the power side-channel is unavailable, they can even be used to break power analysis countermeasures.

Countermeasures against EM attacks on specific implementations fall into two broad categories: ***signal strength reduction*** and ***signal information reduction***. Techniques for signal strength reduction include circuit redesign to reduce egregious unintentional emanations and the use of shielding and physically secured zones to reduce the strength of compromising signals

available to an adversary relative to ambient thermal noise. Techniques for signal information reduction rely on the use of randomization and/or frequent key refreshing within the computation so as to substantially reduce the effectiveness of statistical attacks using the available signals.

5.1.5 Acoustic Attack

Most side-channel attack research has focused on electromagnetic emanations (TEMPEST), power consumption and, recently, diffuse visible light from CRT displays. However, one of the oldest eavesdropping channels, namely acoustic emanations, has received little attention

Very recently, Shamir et al. [6] have demonstrated a preliminary proof-of-concept that a correlation exists between the sound of a processor and its computation. One may consider the approach the P. Wright used in 1965 is likely one of the primitive acoustic attacks. However, this is a relatively new field, and much work needs to be done.

5.1.6 Visible Light Attack

Kuhn demonstrated [7] — via both sophisticated analysis as well as direct experiment — that the average luminosity of a CRT's diffuse reflection off of a wall can sufficient to reconstruct the signal displayed on the CRT (so shielding the CRT to protect against leaking information via electromagnetic radiation may not be sufficient). One outstanding characteristic of this attack is that physical access is not required. Kuhn also speculated that the same techniques are equally applicable to LED signals. Even without line of sight, the adversary may be able to read the signals that a trusted computing platform's optical output channels emit.

Loughry and Umphress [153] described how optical radiation emitted from computer LED (light-emitting diodes) status indicators can be analyzed to infer the data being processed by a device. In [153], a taxonomy of compromising optical emanations is developed, and design changes are described that will successfully block this kind of "Optical TEMPEST" attack.

5.1.7 Error Message Attack

In many standards, e.g. SSL/TLS, IPSEC, WTLS, messages are first pre-formatted, then encrypted in CBC mode with a block cipher. Decryption needs to check if the format is valid. Validity of the format is easily leaked from communication protocols in a chosen ciphertext attack since the receiver usually sends an acknowledgment or an error message. This can become a useful side channel for cryptanalysis and the attack exploiting this side channel is often called error message attack.

Having access to a decryption oracle maybe unrealistic in practice sometime. An adversary, however, can exploit side channels which return enough information about a ciphertext to be decrypted easily. Side channels arise frequently in practice by giving the ability to an adversary to induce predictable changes to plaintexts through modification of the ciphertext. We will review some of these attacks on both symmetric and asymmetric encryption schemes. This attack works as follow: model this behavior as a ***(padding) oracle*** that returns VALID if plaintext is correctly padded, otherwise INVALID.

Vaudenay [144] described error message analysis attacks on symmetric-key encryption when messages are first formatted by padding and then encrypted with a block cipher in CBC mode. In

case of CBC in symmetric schemes, the length of a message must be a multiple of the block length. When this is not the case, padding must be used. What the receiver should do after decryption if he discovers that the padding is not valid depends on the protocol used. If such a padding after decryption is invalid, SSL/TLS [87] specify that the session be torn down, ESP in IPSec [89] just logs the error and WTLS [88] returns an error message. If adversary can ascertain the padding error status, it can use it as a side channel to mount a CCA (chosen cipher attack) attack.

Clearly, several popular padding schemes, which are used today in order to transform block ciphers into variable-input-length encryption schemes, can introduce an important security flaw. Correctness of the plaintext format is indeed a hard-core bit which easily leaks out from the communication protocol. One can really have some insecure standards which use unbroken cryptographic primitives. This was already well known in the public key cryptography world. Vaudenay's results have demonstrated that the situation of symmetric cryptography is virtually the same [144].

Paterson and Yau [151] employed the padding oracle attacking method similar to Vaudenay's to analyse the padding methods of the ISO CBC-mode encryption standard. More recently, Yau et al. at FSE 2005 [152] generalized the padding oracle attack against block ciphers using CBC mode. They considered the security of CBC-mode encryption against padding oracle attacks in secret, random IV setting.

Error message based side channel attacks are not only typical of symmetric systems but of public key systems as well. Assume the attacker has access to an oracle that returns a bit telling whether the ciphertext corresponds to data encrypted according to RSA standard PKCS #1 (v1.5). On the receiving end, receiver parses block from left to right to see if it is PKCS #1 conforming. Using an oracle that tells whether a ciphertext is PKCS#1 conforming, one can break this RSA encryption scheme using about 1 million queries [142,143].

The most prominent and convincing example of side-channel attacks exploiting error messages may be Bleichenbacher's attack [142] on the RSA encryption scheme as specified in the PKCS#1 v1.5 standard. This version of RSA encryption, which specifies a method for formatting the plaintext message prior to application of the RSA function, is widely deployed in practice including in the SSL protocol for secure web communications. For 1024-bit RSA moduli, Bleichenbacher's attack enables an adversary to obtain the decryption of a target ciphertext c by submitting about one million carefullychosen ciphertexts related to c to the victim and learning whether the ciphertexts were rejected or not. The attack necessitated a patch to numerous SSL implementations. The RSA-OAEP encryption scheme was proposed by Bellare and Rogaway [145] and proved secure in the random oracle model by Shoup [146] and Fujisaki et al. [147]. It has been included in many standards including the v2.2 update of PKCS#1. Manger [143] presented his attack on RSA-OAEP in 2001.

After the publication of the results of Bleichenbacher and Manger [142,143], it is widely believed to be important to include a strong integrity check into RSA encryption. The phase between decryption and integrity verification is critical as any leak of information may present a security risk. Version 2 of PKCS #1 introduced a new algorithm RSA-OAEP that uses Optimal Asymmetric Encryption Padding (OAEP) to counteract the previous attack.

Klíma et al. [148] introduced a new side channel attack on a plaintext encrypted by EME-OAEP PKCS#1 v.2.1. What they attacked is the part of the plaintext which is shielded by the OAEP method. They also showed that Bleichenbacher's and Manger's attack on the RSA

encryption scheme PKCS#1 v.1.5 and EME-OAEP PKCS#1 v.2.1 can be converted to an attack on the RSA signature scheme with any message encoding (not only PKCS). A general idea of fault-based attacks on the RSA-KEM scheme was also presented. These attacks would highlight the fact that the RSA-KEM scheme is not an entirely universal solution to problems of RSAES-OAEP implementation and that even here the manner of implementation is significant.

Further more, Klíma et al. [149] pointed out that incorporating a version number check over PKCS#1 plaintext used in the SSL/TLS also creates a side channel that allows an attacker to invert the RSA encryption. Using this attack, one can either recover the premaster-secret or sign a message on behalf of the server in an SSL/TLS session.

Even so, one can also propose adding a cryptographic checkable redundancy code (crypto-CRC) of the whole padded message (like a hashed value) in the plaintext and encrypt

$message \mid padding \mid H(message \mid padding)$, where H is a secure hash function.

In this way, any forged ciphertext will have a negligible probability to be accepted as a valid ciphertext. Basically, attackers are no longer able to forge valid ciphertexts, so the scheme is virtually resistant against chosen ciphertext attacks.

Obviously it is important to pad before hashing: padding after hashing would lead to the a similar attack. The right enciphering sequence is thus $(pad, hash, encrypt)$. Conversely, the right deciphering sequence consists of decrypting, checking the hashed value, then checking the padding value. Invalid hashed value must abort the decipherment.

5.1.8 Cache-based Attack

Previously proposed timing attacks make use of the fact that conditional branches that occur during encryption processing cause variations in encryption time. CPU cache misses, however, can also cause such variations. In this regard, most of the recent computers employ a ***CPU cache***, abbreviated simple to a ***cache*** from here on, between the CPU and main memory, since this type of hierarchical structure can speed program run-time on the average. If, however, the CPU accesses data that were not stored in the cache, i.e. if a cache miss occurs, a delay will be generated, as the target data must be loaded from main memory into the cache. The measurement of this delay may enable attackers to determine the occurrence and frequency of cache misses. This is where the cache-based side channel attacks goes.

The original idea that cache memory could be used as a side-channel which leaks information during the run of a cryptographic algorithm was proposed by Kelsey et al. [67]. The idea was expanded by Page to systematically examine the theoretical use of cache memory as a cryptanalytic side-channel. Such an attacks using side-channel information based on CPU delay against block ciphers are proposed in [43,44]. This is usually classified as a side-channel attack on software-implemented ciphers, and it has already broken MISTY1 [44], DES, AES and Camellia [43] et al. successfully.

Osvik et al. [47] further expanded the idea to the block cipher with substitute and permutation network structure, such as AES. Their attacking method can be used for cryptanalysis of cryptographic primitives that employ data-dependent table lookups. More importantly, they demonstrated an extremely strong type of attack, which requires knowledge of neither the specific plaintexts nor ciphertexts, and works by merely monitoring the effect of the cryptographic process on the cache.

A number of countermeasures to mitigate the cached-based attacks have been proposed, to

name a few, to remove cache or cached S-box access, to disable cache flushing, to perform time and miss skewings, to use application-specific algorithmic masking, to depend on operating system support, to adopt partitioned cache hardware architecture, and so on [45,46,47].

It is claimed that primitives that are normally implemented without lookup tables, such as the SHA family and bitsliced Serpent, are impervious to the attacks described here [47]. Meanwhile, finding an efficient solution that is application- and architecture-independent still remains an open problem.

5.1.9 Frequency-based Attack

C. C. Liu proposed a frequency-based side channel attack against mobile devices such as PDAs, cell phones and pagers [57]. His method is efficient even when traces are misaligned in actual attacking experiments, whereas the previously researched DEMA fails in such condition. In addition, the proposed first-order frequency attack is capable of defeating the desynchronization countermeasure that randomly inserts delays.

However, it may be a pity that the countermeasures against this kind of frequency-based attack are not addressed.

5.1.10 Scan-based Attack

Scan based test is a powerful test technique. However, it is an equally powerful attack tool. In 2004, Yang et al. [58] used scan chains as a side channel to recover secret keys from a hardware implementation of DES.

By using one build-in self-test scheme, the internal status of cryptographic chips will not be scanned out and such scan based attacks can be avoided. Luckily enough, this kind of self-test is already recommended by FIPS 140-2 be the physical security requirement of cryptographic chips [116]. However, it is pointed out here that the high fault coverage of scan based test makes developing a secure scan based solution to cryptographic chips interesting.

5.1.11 Combination of Side Channel Attacks

In [4] , a combination of timing analysis and power analysis was used for theoretical attacks. Such a combination could be used to circumvent some countermeasures against specific side-channel attacks. A simple example for the use of several side-channels is the measurement of the time between significant features in the power trace. More recently, researchers have also examined the potential for multi-channel attacks which utilize multiple side-channels such as power and EM simultaneously [5].

Combinations of other two or more side channels may lead to attacks as well. However, they have not been intensively investigated.

5.1.12 Combination of SCA and Mathematical Attacks

Traditional cryptanalysis techniques can similarly be combined with side-channel attacks to uncover the secret key and/or break the implementation details of the ciphers. In cast of this, even a small amount of side-channel information is sufficient to break common ciphers. For example,

differential fault analysis that uses deliberate injection of faults requires between 50 to 200 cipher text blocks to recover a key of symmetric block cipher DES, while the best traditional attack requires approximately 64 terabytes of plain text and cipher text encrypted under a single key.

There are few fruits in the combination of side channel attacks and traditional mathematical attacks for the moment. Yet, we would like to estimate with a bold hand that this combination might become one of the most devastating attacks against a cryptosystem.

5.1.13 Optimization of Attacks

In a SCA attack, the attacker usually guesses the secret key portion by portion. As a consequence, a large number of measurements are required to mount a successful SCA attack. However in the real world of a SCA attack, the number of measurements is often limited, or it is at least costly to perform a large number of measurements. From the attacker's point of view it is hence desirable to minimize the error probabilities for the guesses of the particular key parts (for a given number of measurements) or vice versa, to minimize the number of measurements which is necessary for a successful attack. If the outcome of the previous guesses has an impact on the guessing strategy of the present key part it is additionally desirable to have criteria with which the correctness of the previous guesses can be verified with reasonable probability.

In order to exploit the side-channel information in an optimal manner, Schindler developed a general approach to optimize the SCA attacks through using stochastic process methods and statistical decision theory [112]. It was demonstrated that by applying appropriate stochastic methods it was possible to increase the efficiency of a number of known attacks considerably, in one case even by factor 50.

Noticing the ample power of the optimizations, we are justified in anticipating that more advanced statistical methods and even multivariate data analyzing theories be adopted to optimize the well-known side channel attacks.

5.2 Concrete Countermeasures

So far, there are many strategies (both in hardware and software) being proposed to combat side-channel attacks, among which some general strategies are [76]:

- de-correlate the output traces on individual runs (e.g., by introducing random timing shifts and wait states, inserting dummy instructions, randomization of the execution of operations, etc.);
- replace critical assembler instructions with ones whose "consumption signature" is hard to analyze, or re-engineer the critical circuitry which performs arithmetic operations or memory transfers;
- make algorithmic changes to the cryptographic primitives so that attacks are provably inefficient on the obtained implementation, e.g., masking data and key with random mask generated at each run.

It had been shown [75,76] that among all these kinds of countermeasures, algorithmic techniques are the most versatile, all-pervasive, and may be the most powerful. Also, in many contexts they are the cheapest to put in place. Software-based countermeasures include introducing dummy instructions, randomization of the instruction execution sequence, balancing Hamming weights of the internal data, and bit splitting. On the hardware level, the

countermeasures usually include clock randomization [80,91], power consumption randomization or compensation [82], randomization of instruction set execution and/or register usage [83]. However, the effect of these countermeasures can be reduced by various signal processing techniques [84]. Software countermeasures against SCA attacks considerably hinder performance of cryptographic algorithms in terms of memory or execution time or both. One of the challenges is to achieve secure implementation with as little extra cost as possible.

Choosing an appropriate resistance level of the countermeasures may depend on the value of your data and the power of adversaries (for example, his knowledge and resources et al.). Evaluating the resistance level should be done at least from the following three angles: adversary's power (including his knowledge, resources and skills et al.), the attack's power (which is closely related to the state-of-the-art) and the countermeasures' effectiveness. It appears that a combination of hardware and software countermeasures yields a very good security/cost ratio.

We can find the following issues and related discussions in the open literatures: attacks, countermeasures (both software and hardware) and theoretical models. Actually these literatures are far from enough for providing us with means of evaluating attacks and designing sound countermeasures. To name a few, as a first approximation, we ignore coupling effects and create a linear model in power analysis attacks, i.e., we assume that the power consumption function of the chip is simply the sum of the power consumption functions of all the events that take place [37]. Actually even very small couplings can provide a rich source of compromising emanations. Exploiting these emanations can be much more effective than trying to work with direct emanations [133]. Actually, a lot of intensive and consistent work should be required in these fundamental research areas.

5.2.1 Randomization

The most general method to counter SCA attacks is to randomize data that may leak through various side channels, such as power consumption, electromagnetic radiation, or execution time. The problem is to guarantee that an attacker may obtain only random information, and thus cannot gain any useful knowledge about the actual initial and/or intermediate data involved in computations.

In case of elliptic curve cryptosystem, randomized projective coordinates method is a practical countermeasure against SCA attacks in which an attacker cannot predict the appearance of a specific value because the coordinates have been randomized. For example, Okeeya et al. proposed an SCA-resistant scalar multiplication method that is allowed to take any number of pre-computed points [12]. The proposed scheme essentially intends to resist the simple power analysis, not the differential power analysis.

The standard DPA utilizes the correlation function that can distinguish whether a specific bit is related to the observed calculation. In order to resist DPA, we need to randomize the parameters of elliptic curves. There are three standard randomizations [13,14] commonly available today: (1) the base point is masked by a random point; (2) the secret scalar is randomized with multiplier of the order of the curve; and (3) the base point is randomized in the projective coordinate (or Jacobian coordinate). Some attacks or weak classes against each countermeasure have been proposed [25,18]. However, if these randomization methods are simultaneously used, no attack is known to break the combined scheme. In other words, SPA-resistant schemes can be easily converted to be DPA-resistant ones using these randomizations.

On the contrary, there still appear some schemes which try to achieve the SPA- and DPA-resistance simultaneously without using the combinations, e.g. randomized window methods [26,27,15,28], etc.

5.2.2 Blinding

Blinding is originally a concept in cryptography that allows a client to have a provider compute a mathematical function $y = f(x)$, where the client provides an input x and retrieves the corresponding output y, but the provider would neither learn x nor y. This concept is useful if the client cannot compute the mathematical function f all by himself, for example, because the provider uses an additional private input in order to compute f efficiently.

The first blinding technique was proposed by Chaum as part of the Chaum blind signature [40,41]. It is based on a homomorphic property of the RSA signing function. Blinding techniques are also the most effective countermeasure against remote timing analysis of web servers [39] and against power analysis and/or timing analysis of hardware security modules.

5.2.3 Masking

The data masking technique is the most widely used countermeasure against power analysis and timing attacks at a software level. Masking an algorithm means masking the intermediate values which are processed in the computation of the algorithm operation.

Data masking is also one of the most powerful software countermeasures against side channel attacks [75,85]. The idea is very simple: the message and the key are masked with some random masks at the beginning of computations, and thereafter everything is almost as usual. Of course, the value of the mask at the end of some fixed step (e.g., at the end of the round or at the end of a linear part of computations) must be known in order to re-establish the expected data value at the end of the execution; we call this ***mask correction***.

In case of AES, this countermeasure means making the intermediate bytes processed in an AES computation. Masking a byte value x mean to choose a random m (***the mask***) and to define a function f (the ***masking***) which takes both values as input to calculate the masked output: $f(x,m)=x*m$. The operator * is either defined as bit-wise XOR operation , denoted by $+$, (***additive masking***), or as multiplication, denoted by $\times$, over a finite field (***multiplicative masking***).

In [72], Trichina et al. proposed an optimized countermeasure for the AES block cipher consisting in transforming a boolean mask to a multiplicative mask prior to a non-linear Byte Substitution operation (thus, avoiding S-box re-computations for every run or storing multiple S-box tables in RAM), while preserving a boolean mask everywhere else (***adaptive masking method***).

6. Possible Impacts on Cryptographic Module Security Testing

From the detailed discussions above, a conclusion can be easily drawn that SCA attacks pose a serious and real threat on the security of cryptographic modules. Consequently, it is supposed to come to a common understanding that not only the designer or the implementor but also the connor of a cryptographic module should be completely and clearly aware of the tremendous hazards in which the passive attacks result. In this section, we will discuss some possible impacts

of SCAs on the cryptographic module security testing.

The traditional black-box method widely used in the design and analysis in the cryptosystem has server limitations, therefore, will not be fitted with the current advancements and developments of the information security engineering. More other aspects should be included in the evaluation of a system in order to justify the overall security more accurately and objectively. It is required that at least the secure implementation of cryptographic modules (protocols / systems) be seriously taken into consideration when designing a functional cryptographic component.

Choosing an appropriate cryptographic module or device still comes down to the level of security you need. Namely, classified testing and evaluation is not only practical but also necessary. During this complex and independent testing process, new security modeling theory and security testing method are required.

Not all parts of a system are susceptible for SCA attacks, but it is not a trivial task to specify which parts have to be secured and which are not. To reduce the risk of unprotected susceptible parts, expensive countermeasures are often applied for parts that are actually not susceptible; therefore, all countermeasures should be designed and implemented with a proper and quantitative evaluation of their effects on the overall effectiveness. Following this observation, a natural problem comes out that whether it is possible to develop some methodologies to allow SCA simulations in early design stages of the system or not.

Besides the current security testing requirements, the requirements of standard FIPS 140 should be extended to a larger scale and to cover the following aspects: analysis of cryptographic protocols; analysis of effectiveness of key management; analysis of side channels or similar vulnerabilities; analysis of correct use of the cryptographic module in a larger product; any statements about non-FIPS approved or FIPS allowed algorithms; and so on.

To solely enlarge the key size, in some cases, will not necessarily or apparently increase the security level. In [70,71], the authors argued that “the longer the key length, the easier the attacks becomes”. Even what was being attacked in [70] is actually an RSA implementation using sliding window method, the success of the attack still reveals some meaningful things. Increasing key length is a standard countermeasure to cryptanalysis. However, longer key length generally means greater side channel leakage. For embedded RSA crypto-systems the increase in leaked data outstrips the increase in secret data so that, in contrast to the improved mathematical strength, longer keys may, in fact, lead to lower security [71].

Attacking and designing for security is on the whole not balanced to each other with regards to real-world application scenario. In term of this, in our own opinion, there is probably a gap between the evaluation (or testing) and the attacks. Attacks sometimes can be ad-hoc in nature; they seize on a particular aspect of the target system and exploit it. It can be helpful to have generalized attacks, but it is not really necessary. Attacks also do not require rigorous analysis; a simple demonstration of the efficacy of an attack is sufficient to cast doubt on the security of a system, even in the absence of an explanation of the very details of the attack’s mechanism. Designing for security, on the other hand, requires both generality and rigor.

Other than proposing the concrete metrics for evaluation, we would just suggest some security requirements that a cryptographic module security testing standard should specify as follows.

- Any criteria must be testable;
- The criteria should be able to evaluate the performance of different implementations so

as to seek out the most efficient one;

- The criteria should be able to evaluate the cost of the implementations, such as memory cost and manufacture cost.
- Any criteria must be easy to deploy;

As a simple fact, most side-channel attacks are not covered currently by almost all security models of the theoretical cryptography. Hence, even the (implementations of) provable secure cryptographic algorithms may be attacked due to some possible information leakages. Actually, there are already some published provable secure cryptographic schemes broken under the SCA attacks. It is worth noting that the fact SCA attacks have been successfully mounted against semantically secure cryptosystems does not undermine the mathematical base of these cryptosystems, nor reflect the inherent mathematical weaknesses in these cryptosystems.

7. Conclusion

Cryptology may be seen as a continuous struggle between cryptographers and cryptanalysts. Attacks on cryptography have an equally long history. The security of cryptographic modules for providing a practical degree of protection against white-box (total access) attacks should be examined in a totally un-trusted execution environment.

As Dr. Bruce Schneier already pointed out in 1998 that [48], "Strong cryptography is very powerful when it is done right, but it is not a panacea. Focusing on cryptographic algorithms while ignoring other aspects of security is like defending your house not by building a fence around it, but by putting an immense stake in the ground and hoping that your adversary runs right into it". Nowadays, this argument should be further revised. Building a fence around the house is already far from enough, who can guarantee that the attacker will not dig a tunnel under the fence to bypass the defense (for example, imagine the famous tunnel warfare during the anti-Japanese war in China)? One may attempt to make the ground as hard as possible, yet who can guarantee that your opponent will fall abruptly from the sky someday? Bear in mind that your opponent is no less clever than you at all. Probably, you can never ascertain what your opponent will do next.

We have surveyed side-channel attacks and the relevant countermeasures. A wide array of countermeasures against side channel attacks have been developed by researchers to provide the protections. We believe that a clear understanding of attacks as well as the trade-offs associated with deploying countermeasures will enable a system architect to develop a truly secure system.

Prof. Bart Preneel once stated that "the 'crypto problem' is not solved; many challenging problems are ahead of both research community and industry, with the secure and efficient implementation of cryptographic schemes being included". The broadness of the range of possible attack avenues complicates the task of addressing them. Up till now, at least ten kinds of side channels have been completely or partially lifted their veils. One is justified in estimating that more and more side channels will be discovered and then will likely be exploited by the adversary to mount an attack. What will be the next side channel and what will be the last one? Therefore, the design of implementations of a cryptographic module or system requires a strong awareness of the potential implementation weaknesses that would become security flaws, and careful consideration of security during all aspects of the architecture, hardware, and software design processes. It is the very time that the resistivity of cryptographic modules against side-channel attacks be correctly evaluated in the security testing procedure of such modules. What we are waiting for?

There are also many interesting topics for the academic researcher. In particular, the rigorous security models of side-channel attacks are still poorly understood, and there may be considerable scope for applying formal techniques and computational complexity methods to their analysis. It must not be forgotten that there are still some side channels waiting to be discovered.

Most of the problems that we have discussed relating to SCA and the countermeasures will appear also in relation to the other systemic parameters, thus requiring the study of interaction between a multitude of systemic parameters. One might hope that eventually a general theory and associated methods and tools might emerge that will support the security engineer. All in all, it is important to do both theoretical and practical work in order to get new ideas for these attacks.

Finally, the most important conclusion from this paper is that it is not only a necessity but also a must, in the coming version of FIPS 140-3 standard, to evaluate cryptographic modules for their resistivity against SCA attacks.

Reference

[1] J. Black, H. Urtubia. *Side-channel attacks on symmetric encryption schemes: the case for authenticated encryption*. Proc of 11th USENIX Security Symposium, pp.327-338, 2002.

[2] R. Anderson, M. Kuhn. *Tamper resistance—a cautionary note*. Proc of the 2nd USENIX Workshop on Electronic Commerce,pp.1-11, 1996.

[3] R. Anderson, M. Kuhn. *Low cost attacks on tamper resistant devices*. Proc of the 1997 Security Protocols Workshop,pp.125-136,LNCS 1361, 1997.

[4] W. Schindler. *A Combined Timing and Power Attack*. PKC 2002,LNCS 2274,pp.263-279, 2002.

[5] D. Agrawal, J.R. Rao, P. Rohatgi. *Multi-channel Attacs*. CHES 2003,LNCS 2779,pp.2-16, 2003.

[6] A. Shamir, E.Tramer. *Acoustic cryptanalysis: on nosy people and noisy machines*. Eurocrypt 2004 rump session, 2004.

[7] M. Kuhn. *Optical Time-Domain Eavesdropping Risks of CRT Displays*. Proc of the 2002 Symposium on Security and Privacy, pp.3-18,2002.

[8] 3GPP TS 35.205(V4.0.0). 3rd Generation Partnership Project;Technical Specification Group Services and System Aspects;3G Security;Specification of the MILENAGE Algorithm Set,April 2001.

[9] ETSI SAGE 3GPP AF Task Force. *Report on the design and evaluation of 3GPP Authentication and Key Generation Functions*.

[10] K. Okeya, K. Miyazaki, K. Sakurai. *A Fast Scalar Multiplication Method with Randomized Projective Coordinates on a Montgomery-Form Elliptic Curve Secure against Side Channel Attacks*. ICICS 2001,LNCS 2288,pp.428-439,2002.

[11] C.H. Gebotys, R.J. Gebotys. *A Framework for Security on NoC Technologies*. Proc of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI'03),2003.

[12] K. Okeya, T. Takagi. *A More Flexible Countermeasure against Side Channel Attacks Using Window Method*. CHES'2003,LNCS 2779,pp.397-410,2003.

[13] J.S. Coron. *Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems*. CHES'99,LNCS 1717, pp.292-302,1999.

[14] M. Joye,C. Tymen. *Protections against differential analysis for elliptic curve cryptography: An algebraic approach*. CHES'2001,LNCS 2162,pp.377-390, 2001.

[15] P.Y. Liardet, N.P Smart. *Preventing SPA/DPA in ECC systems using the Jacobi form*. CHES 2001, LNCS 2162,pp.391-401,2001.

[16] E. Brier, M. Joye. *Weierstrass Elliptic Curves and Side-Channel Attacks.* PKC 2002, LNCS 2274,pp.335-345,2002.

[17] M. Joye, J.J. Quisquater. *Hessian elliptic curves and side-channel attacks.* CHES 2001, LNCS 2162, pp.402-410,2001.

[18] Okeya, K., Sakurai, K., *Power Analysis Breaks Elliptic Curve Cryptosystems even Secure against the Timing Attack*. INDOCRYPT 2000, LNCS1977,pp.178-190,2000.

[19] K. Okeya, T. Takagi. *The Width-w NAF Method Provides Small Memory and Fast Elliptic Scalar Multiplications Secure against Side Channel Attacks.* CT-RSA 2003, LNCS 2612,pp.328-342,2003.

[20] W. Fischer, C. Giraud, E.W. Knudsen, J.P. Seifert. *Parallel scalar multiplication on general elliptic curves over Fp hedged against Non-Differential Side-Channel Attacks.* Available at http://eprint.iacr.org/2002/007/.

[21] T. Izu, T. Takagi. *A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks.* PKC 2002, LNCS 2274,pp.280-296,2002.

[22] B. Möller. *Securing Elliptic Curve Point Multiplication against Side-Channel Attacks.* ISC 2001, LNCS 2200, pp.324-334,2001.

[23] B. Möller. *Securing elliptic curve point multiplication against side-channel attacks, addendum: Efficiency improvement*. Available at http://www.informatik.tudarmstadt.de/TI/Mitarb-eiter/moeller/ecc-scaisc01.pdf, (2001).

[24] B. Möller. *Parallelizable Elliptic Curve Point Multiplication Method with Resistance against Side-Channel Attacks.* ISC 2002, LNCS 2433,pp.402-413,2002.

[25] L. Goubin. *A Refined Power-Analysis Attack on Elliptic Curve Crypto -systems.* PKC 2003, LNCS 2567,pp.199-211, 2003.

[26] C.D. Walter. *Seeing through Mist Given a Small Fraction of an RSA Private Key.* CT-RSA 2003, LNCS 2612,pp.391-402,2003.

[27] E. Oswald, M. Aigner. *Randomized Addition-Subtraction Chains as a Countermeasure against Power Attacks.* CHES 2001, LNCS 2162,pp.39-50,2001.

[28] K. Itoh, J. Yajima ,M. Takenaka,N. Torii.*DPA Countermeasures by improving the Window Method.* CHES 2002, LNCS 2523,pp.318-332,2002.

[29] C.H. Lim. *A New Method for Securing Elliptic Scalar Multiplication Against Side-Channel Attacks*. ACISP 2004, LNCS 3108, pp.289-300, 2004.

[30] R.M. Sommer. *Smartly analyzing the simplicity and the power of simple power analysis on smartcards.*CHES 2000, LNCS 1965,pp.78-92,2000.

[31] T.S. Messerges, E.A. Dabbish, R.H. Sloan, *Examining smart-card security under the threat of power analysis attacks.* IEEE Trans. Computers, 51(5), pp.541-552, 2002.

[32] T.S. Messerges, E.A. Dabbish, R.H. Sloan. *Power analysis attacks of modular exponentiation in smart cards*. CHES'99, LNCS 1717,pp.144-157,1999.

[33] R. Novak. *SPA-based adaptive chosen-ciphertext attack on RSA implementation*. PKC 2002, LNCS 2274,pp.252-262,2002.

[34] C.D. Walter. *Sliding windows succumbs to big mac attack*. CHES 2001, LNCS 2162, pp.286-299,2001.

[35] J.C. Yoon, S.W. Jung,S. Lee. *Architecture for an Elliptic Curve Scalar Multiplication*

Resistant to Some Side-Channel Attacks.ICISC 2003, LNCS 2971, pp. 139-151, 2004.

[36] M. Chechik, J. Gannon. *Automatic Analysis of Consistency Between Implementations and Requirements: A Case Study*. Proc of Symposium on Computer Assurance (COMPASS'95), pp.123-132, Gathersburg, MD,June 1995

[37] S. Chari, C. Jutla, J. R. Rao,P. Rohatgi. *A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards*. Proc of 2nd Advanced Encryption Standard (AES) Candidate Conference, Rome, Italy, 1999.

[38] S. Chiyangwa,M. Kwiatkowska. *A Timing Analysis of AODV*. FMOODS 2005, LNCS 3535, pp. 306–321, 2005.

[39] E. Brier, M. Joye. *Weierstra β Elliptic Curves and Side-Channel Attacks*. PKC 2002, LNCS 2274, pp.335-345, 2002.

[40] D. Chaum. *Blind Signatures for untraceable payments*. CRYPTO'82, pp.199-203,1983.

[41] D. Chaum. *Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms.*AUSCRYPT'90, LNCS 453, pp.246-264,1990.

[42] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons, 2001.

[43] Y. Tsunoo, E. Tsujihara, K. Minematsu, H. Miyauchi. *Cryptanalysis of Block Ciphers Implemented on Computers with Cache*. ISITA 2002, 2002.

[44] Y. Tsunoo, T. Saito, T. Suzaki, M. Shigeri, H. Miyauchi.*Cryptanalysis of DES Implemented on Computers with Cache*. CHES 2003, LNCS 2779, pp.62–76, 2003.

[45] D. Page. *Defending against cache-based side-channel attacks*. Information Security Technical Report,Vol.8, No.1,2003.

[46] D. Page. *Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel*. Available at http://eprint.iacr.org/2002/169.pdf.

[47] D.A. Osvik, A. Shamir, E. Tromer. *Cache Attacks and Countermeasures: the Case of AES*. Available at http://eprint.iacr.org/2005/271.pdf.

[48] B. Schneier. *Cryptographic Design Vulnerabilities*. IEEE Computer, Vol.31, No.9,pp.29-33,1998.

[49] P. Kocher. *Timing attacks on implementations of Diffie-Hellmann, RSA, DSS,and other systems*.CRYPTO'96, LNCS 1109,pp.104-113,1996.

[50] D.X. Song, D. Wagner, X. Tian. *Timing Analysis of Keystrokes and Timing Attacks on SSH*. Proc of 10th USENIX Security Symposium, 2001.

[51] J. Cathalo, F. Koeune, J.J. Quisquater. *A New Type of Timing Attack: Application to GPS*. CHES 2003,LNCS 2779,pp.291-303,2003.

[52] K. Sakurai, T. Takagi. *A Reject Timing Attack on an IND-CCA2 Public-key Cryptosystem*. ICISC 2002,LNCS 2587, pp.359-374,2003.

[53] B.N. Levine, M.K. Reiter, C. Wang, M. Wright. *Timing Attacks in Low-Latency Mix Systems*. FC 2004, LNCS 3110, pp.251-265, 2004.

[54] D. Brumley, D. Boneh. *Remote Timing Attacks are Practical*. Proc of 12th Usenix Security Symposium,2003.

[55] J.F. Dhem, F. Koeune, P.A. Leroux, P. Mestre, J.J. Quisquater, J.L. Willems. *A practical implementation ofthe timing attack*. Proc. of CARDIS 1998, 1998.

[56] M. Buchholtz, S.T. Gilmore, J. Hillston, F. Nielson. *Securing Statically-verified Communications Protocols Against Timing Attacks*. Electronic Notes in Theoretical Computer

Science,128(4),pp.123-143,2005.

[57] C.C. Tiu. *A New Frequency-Based Side Channel Attack for Embedded Systems*. Master degree thesis, Deparment of Electrical and Computer Engineering,University of Waterloo,Waterloo, Ontario, Canada, 2005.

[58] B. Yang, K. Wu, R. Karri. *Scan-based Side-Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard*. Proc of International Test Conference 2004(ITC'2004),Charlotte,2004.

[59] P. Kocher, J. Jaffe, B. Jun. *Differential power analysis*. CRYPTO'99,LNCS 1666,pp.388-397,1999.

[60] J.I. Hartog, E.P. Vink. *Virtual Analysis and Reduction of Side-Channel Vulnerabilities of Smartcards*. Proc of FAST 2004,2004.

[61] E.P. Antoniadis, A.G. Voyiatzis, D.N. Serpanos,A. Traganitis. *Software Siumulation of Active Attacks in Cryptographic Systems*.Technical Reoprt: CSD-TR-2001-01.University of Crete,2001.

[62] P.A. Fouque,F. Valette. *The Doubling Attack – Why Upwards Is Better than Downwards*. CHES 2003, LNCS 2779, pp.269-280, 2003.

[63] S. Ravi,A. Raghunathan,S. Chakradhar. *Tamper Resistance Mechanisms for Secure Embedded Systems*. Proc of the 17th International Conference on VLSI Design(VLSID'04), pp.605-611,2004.

[64] P. Kocher, R. Lee, G. McGraw, A. Raghunathan, S. Ravi. *Security as a New Dimension in Embedded System Design*. Proc of IEEE DAC'2004, pp.753-761,California,USA,2004.

[65] C. Giraud, H. Thiebeauld. *A survey of fault attacks*. Proc of CARDIS 2004,pp.159-176, Kluwer Press, 2004.

[66] K. Tiri, I. Verbauwhede. *A VLSI design flow for secure side-channel attack resistant ICs*. Proc of the Design, Automation and Test in Europe Conference and Exhibition(DATE'05),pp.1530-1591,2005.

[67] J. Kelsey,B. Schneier,D. Wagner,C. Hall. *Side channel cryptanalysis of product ciphers*. Proc of 5th European Symposium on Research in Computer Security, LNCS 1485,pp.97–110,1998.

[68] J. Lano,N. Mentens,B. Preneel, I. Verbauwhede. *Power analysis of synchronous stream ciphers with resynchronization mechanism*.

[69] C. Rechberger. *Side channel analysis of stream ciphers*. Master's Thesis. IAIK,Graz University of Technology,Austria,2004.

[70] C.D. Walter. *Sliding Windows Succumbs to Big Mac Attack*. CHES 2001, LNCS 2162, pp.286-299, 2001.

[71] C.D. Walter. *Longer keys may facilitate side channel attacks* .SAC 2003, LNCS 3006, pp. 42-57, 2004.

[72] E. Trichina, D.D. Seta, L. Germani. *Simplified Adaptive Multiplicative Masking for AES*. CHES 2002, LNCS 2523, pp.187-197, 2003.

[73] E. Oswald, S. Mangard, N. Pramstaller. *Secure and Efficient Masking of AES - A Mission Impossible?* . Technical Report IAIK - TR 2003/11/1.

[74] E. Trichina,L. Korkishko. *Secure and Efficient AES Software Implemen- tation for Smart Cards*. WISA 2004,LNCS 3325, pp.425-439, 2004.

[75] S. Chari, C. Jutla, J. Rao, P. Rohatgi. *Towards sound approaches to counteract*

power-analysis attacks. CRYPTO'99, LNCS 1666,pp398-412,1999.

[76] L. Goubin,J. Patarin.*DES and differential power analysis*.CHES'99, LNCS 1717,pp.158-172,1999.

[77] T. Messerges. *Securing the AES finalists against power analysis attacks*. FSE 2000,LNCS 1978,pp.150-165,2000.

[78] K. Okeya,T. Takagi. *Security Analysis of CRT-Based Cryptosystems*. ACNS 2004, LNCS 3089, pp.383-397, 2004.

[79] E. Trichina, T. Korkishko. *Secure AES Hardware Module for Resource Constrained Devices*. ESAS 2004, LNCS 3313, pp.215-229,2005.

[80] P. Kocher, J. Jaffe,B. Jun. *Using unpredictable information to minimize leakage from smartcards and other cryptosystems*. USA patent, International Publication number WO 99/63696,1999.

[81] E. Sprunk. *Clock frequency modulation for secure microprocessors*. USA patent number WO 99/63696,1999.

[82] S. Fruhauf,L. Sourge. *Safety device against the unauthorized detection of protected data*. U.S. patent 5,404,402,1995.

[83] D. May, L.H. Muller, N.P. Smart. *Random register renaming to foil DPA*. CHES 2001, LNCS 2162, pp.28-38,2001.

[84] C. Clavier, J.S. Coron, N. Dabbous. *Differential Power Analysis in the Presence of Hardware Countermeasures*. CHES 2000,LNCS 1965, pp.252-263,2000.

[85] L. Goubin. *A sound method for switching between boolean and arithmetic masking*. CHES 2001,LNCS 2162,pp.3-15,2001.

[86] J.Blömer, J. Guajardo, V. Krummel. *Provably Secure Masking of AES*. SAC 2004, LNCS 3357, pp.69-83, 2005.

[87] http://www.openssl.org

[88] Wireless Transport Layer Security. Wireless Application Protocol WAP-261-WTLS-200 -10406-a. Wireless Application Protocol Forum, 2001.http://www.wapforum.org/

[89] S. Kent, R. Atkinson. *Security Architecture for the Internet Protocol*. RFC 2401,standard tracks, the Internet Society, 1998.

[90] D. Boneh, R.A. DeMillo, R.J. Lipton. *On the importance of checking cryptographic protocols for faults*. EUROCRYPT '97,LNCS 1233,pp.37-51,1997.

[91] M. Joye, A.K. Lenstra, J.J. Quisquater. *Chinese remaindering cryptosystems in the presence of faults*. Journal of Cryptology,Vol.12,pp.241-245,1999.

[92] E. Biham, A. Shamir. *Differential fault analysis of secret key cryptosystems*. CRYPTO '97,LNCS 1294,pp.513-525,1997.

[93] I. Biehl, B. Meyer, V. Műller. *Differential fault attacks on elliptic curve cryptosystems*. CRYPTO 2000,LNCS 1880,pp.131–146,2000.

[94] R. Karri, K. Wu, P. Mishra, Y. Kim. *Concurrent error detection of faultbased side-channel cryptanalysis of 128-bit symmetric block ciphers*.Proc. of IEEE Design Automation Conference,pp.579-585,2001.

[95] S. Skorobogatov, R. Anderson. *Optical Fault Induction Attacks*. CHES 2002, LNCS 2523, pp.2-12,2003..

[96] S.M. Yen, M. Joye. *Checking before output may not be enough against fault-based cryptanalysis*. IEEE Trans. on Computers,Vol.49,pp.967-970,2000.

[97] S.M. Yen, S.J. Kim, S.G. Lim, S.J. Moon. *RSA Speedup with Residue Number System immune from Hardware fault cryptanalysis*. ICICS 2001,LNCS 2288,pp.397-413,2002.

[98] M. Ciet, C. Giraud. *Transient Fault Induction Attacks on XTR*. ICICS 2004,LNCS 3269,pp.440-451,2004.

[99] G. Piret, J.J. Quisquater. *A Differential Fault Attack Technique Against SPN Structures, with Application to the AES and Khazad*. CHES 2003,LNCS 2779,pp.77-88,2003.

[100] J.J. Hoch, A. Shamir. *Fault Analysis of Stream Ciphers*. CHES 2004,LNCS 3156,pp.240-253,2004.

[101] D. Wagner. *Cryptanalysis of a Provably Secure CRT-RSA Algorithm*. Proc of ACM CCS'04, Washington, DC, USA,2004.

[102] D. Naccache, P.Q. Nguy, M. Tunstall, C. Whelan. *Experimenting with Faults, Lattices and the DSA*. PKC 2005,LNCS 3386,pp.16-28,2005.

[103] C. Giraud, E.W. Knudsen. *Fault Attacks on Signature Schemes*. ACISP 2004, LNCS 3108, pp.478-491,2004.

[104] F. Bao, R.H. Deng, Y. Han, A. Jeng, A.D. Narasimhalu, T. Ngair. *Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults*. SP'98,LNCS 1361,pp.115-124,1998.

[105] S. Hariri, G. Qu,T. Dharmagadda,M. Ramkishore,C.S. Raghavendra. *Impact Analysis of Faults and Attacks in Large-Scale Networks*. IEEE Security & Privacy,Vol.3,pp.1540-7993,2003.

[106] C.S. Laih, F.K. Tu, Y.C. Lee. *On the Implementation of Public Key Cryptosystems against Fault-Based Attacks*. IEICE Trans. on Fundamentals. Vol.E82-A,No.6,pp.1082-1089,1999.

[107] T. Sumi, O. Mizuno, T. Kikuno, M. Hirayama. *An Effective Testing Method for Hardware Related Fault in Embedded Software*. IEICE Trans. on Information & System,Vol.E88-D,No.6,pp.1142-1149,2005.

[108] S.M. Yen, S. Moon, J.C. Ha. *Hardware Fault Attack on RSA with CRT Revisited*. ICISC 2002,LNCS 2587,pp.374-388,2003.

[109] J. Blömer, J.P. Seifert. *Fault Based Cryptanalysis of the Advanced Encryption Standard (AES)*. FC 2003,LNCS 2742,pp.162-181,2003.

[110] D. Boneh, R.A. DeMillo, R.J. Lipton. *On the Importance of Eliminating Errors in Cryptographic Computations*. Journal of Cryptology, Vol.14, pp.101-119, 2001.

[111] R. Anderson, M. Bond, J. Clulow, S. Skorobogatov. *Cryptographic Processors - A Survey*. IEEE Special Issue, 2005.(to appear)

[112] W. Schindler. *On the optimization of side-channel attacks by advanced stochastic methods*. PKC 2005, LNCS 3386,pp.85-103,2005.

[113] P. Wright. *Spy Catcher: The Candid Autobiography of a Senior Intelligence Officer*. Viking Press,1987.

[114] B. Schneier. *Security pitfalls in cryptography*. Available at http://www.s-chneier.com/essay-pitfalls.html.

[115] A. J. Menezes, P. C. Oorschot, S. A. Vanstone. *Handbood of applied cryptography*(5^{th} edition). CRC Press, 2001.

[116] FIPS PUB 140-2 Security Requirements for Cryptographic Modules. available at http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

[117] J.I. Hartong, E.P. Vink. *Virtual Analysis and Reduction of Side-Channel Vulnerabilities of Smartcards*.Proc of FAST 2004, 2004.

[118] Boza Barak. *Non-Black-Box Techniques in Cryptography*. PhD thesis, ISA, 2004.

[119] 3GPP ETSI TR 133-909(V4.0.1). Universal Mobile Telecommunications System (UMTS);3G Security;Report on the design and evaluation of the MILENAGE algorithm set; 3GPP TR 33.909 version 4.0.1 Release 4, June 2001.

[120] http://www.scard-project.org/

[130] http://www.ecrypt.eu.org/

[131] https://www.trustedcomputinggroup.org/

[132] E. Oswald. *On Side-Channel Attacks and the Application of Algorithmic Countermeasures*, PhD dissertation, 2004.

[133] D. Agrawal, B. Archambeault, J.R. Rao, P. Rohatgi. *The EM Side–Channel(s)*. CHES 2002,LNCS 2523,pp.29-45,2003.

[134] Y. Ishai,A. Sahai,D. Wagner. *Private Circuits: Securing Hardware against Probing Attacks*. CRYPTO 2003, LNCS 2729, pp.463-481, 2003.

[135] K. Gandolfi,C.Mourte,F. Olivier. *Electromagnetic Analysis: Concrete Results*. CHES 2001,LNCS 2162,pp.251-261, 2001.

[136] National Security Agency. *NACSIM 5000 Tempest Fundamentals (U)*. Fort George G. Meade, Maryland, USA. Available from http://cryptome.org/nacsim-5000.htm.

[137] J.J. Quisquater, D. Samyde. *Electromagnetic analysis (EMA): measures and countermeasures for smart cards*. E-smart 2001,LNCS 2140,pp.200–210,2001.

[138] M.G. Kuhn, R.J. Anderson. *Soft tempest: hidden data transmission using electromagnetic emanations*. Information Hiding 1998,LNCS 1525,pp.124-142,1998.

[139] E.D. Mulder, P. Buysschaert, S.Börs, P. Delmotte, B. Preleel, G. Vandenbosch, I. Verbauwhede. *Electromagnetic Analysis Attack on a FPGA Implementation of an Elliptic Curve Cryptosystem*. Katholieke Universiteit Leuven, Leuven, Belgium.January 2005.

[140] V. Carlier,H. Chabanne,E. Dottax,H. Pelletier. *Electromagnetic Side Channels of an FPGA Implementation of AES*. Available at http://eprint.iacr.org/2004/145.pdf.

[141] 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, University of Waterloo Waterloo, Canada,2005.

[142] D. Bleichenbacher. *Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1*. CRYPTO'98, LNCS 1462, pp.1-12, 1998.

[143] J. Manger. *A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1 v2.0*. CRYPTO 2001,LNCS 2139,pp.230-238,2001.

[144] S. Vaudenay. *Security Flaws Induced by CBC Padding – Applications to SSL, IPSEC, WTLS* .EUROCRYPT 2002, LNCS 2332, pp.534-545,2002.

[145] M. Bellare,P. Rogaway. *Optimal Asymmetric Encryption*. EUROCRYPT 1994, LNCS 950,pp.92-111,1994.

[146] V. Shoup. *OAEP reconsidered*. Journal of Cryptology,Vol.15,pp.223-249,2002.

[147] E. Fujisaki,T. Okamoto,D. Pointcheval,J. Stern. *RSA-OAEP is secure under the RSA assumption*. CRYPTO 2001,LNCS 2139,pp. 260-274, 2001.

[148] V. Klíma,T. Rosa.*Further results and considerations on side channel attacks on RSA*. CHES 2002,LNCS 2523,pp.244-259,2002.

[149] V. Klíma,O. Pokorný,T. Rosa. *Attacking RSA-Based Sessions in SSL/TLS*. CHES 2003,LNCS 2779,pp.426-440,2003.

[150] V. Klíma,T. Rosa. *Side Channel Attacks on CBC Encrypted Messages in the PKCS#7 Format*. Available at http://eprint.iacr.org/2003/098.pdf.

[151] K.G. Paterson,A. Yau. *Padding Oracle Attacks on the ISO CBC Mode Padding Standard*.CT-RSA 2004,LNCS 2964,,pp.305–323,2004.

[152] A.K.L. Yau,K.G. Paterson,C.J. Mitchell. *Padding Oracle Attacks on CBC-Mode Encryption with Secret and Random IVs*. FSE 2005,LNCS 3557,pp.299-319, 2005.

[153] J. Loughry, D. Umphress. *Information leakage from optical emanations*.ACM Transactions on Information and System Security, Vol.5,pp.262–289, 2002.

[154] I.F Blake, G. Serroussi, N.P. Smart. Advances in elliptic curve cryptography. Cambridge University Press, 2005.

[155] W. Schindler. *A timing attack against RSA with the Chinese Remainder Theorem*. CHES 2000,LNCS 1965,pp.109-124,2000.

[156] A. Hevia,M. Kiwi. *Strength of two data encryption standard implementations under timing attacks*. ACM Transactions on Information and System Security,Vol.2,pp416-437,1999.

[157] B. Canvel,A. Hiltgen,S. Vaudenay,M. Vuagnoux. *Password Interception in a SSL/TLS Channel*. CRYPTO 2003,LNCS 2729,pp.583-599,2003.

[158] NESSIE Project. http://www.cryptonessie.org/

[159] E. Oswald. *Enhancing simple power-analysis attacks on elliptic curve cryptosystems*. CHES 2002,LNCS 2523,pp.82-97,2003.

[160] M. Akkar,R. Bevan,P. Dischamp,D. Moyart. *Power analysis, what is now possible....*ASIACRYPT 2000,LNCS 1976,pp.489-502,2000.

[161] C.H. Gebotys, R.J. Gebotys. *Secure elliptic curve implementations: an analysis of resistance to power-attacks in a DSP processor*. CHES 2002,LNCS 2523,pp.114-128,2003.

[162] D. May, H. Muller,N. Smart. *Non-deterministic processors*. ACISP 2001,LNCS 2119,pp.115-129,2001.

[163] A. Shamir. *Protecting smart cards from passive power analysis with detached power supplies*.CHES 2000,LNCS 1965,pp.71-77,2000.

[164] T. Izu,T. Takagi. *Exceptional procedure attack on elliptic curve cryptosystems*. PKC 2003, LNCS 2567,pp.224-239,2003.

[165] Y. Hitchcock,P. Montague. *A new elliptic curve scalar multiplication algorithm to resist simple power analysis*. ACISP 2002,LNCS 2384,pp.214-225,2002.

[166] T. Izu,B. Möller,T. Takagi. *Improved elliptic curve multiplication methods resistant against side channel attacks*. INDOCRYPT 2002,LNCS 2551,pp.296-313, 2002.

[167] E. TRICHINA AND A. BELLEZZA. *Implementation of elliptic curve cryptography with built-in counter measures against side channel attacks*. CHES 2002,LNCS 2523,pp.98-113, 2002.

[168] M. Hasan. *Power analysis attacks and algorithmic approaches to their countermeasures for Koblitz curve cryptosystems*. IEEE Transactions on Computers,Vol.50,pp.1071-1083,2001.

[169] S.W. Smith. *Trusted computing platforms: design and applications*. Springer Press, 2005.