CHAPTER 3: ELECTRICAL AND ELECTRONIC ENGINEERING

Security in Embedded Systems: Design Challenges

Peter.D. Dawoud [1], D.S. Dawoud [2], Adronis Niyonkuru[3]

[1]PhD student, Waterloo University Canada
[2]Professor, Faculty of Applied Sciences, National University of Rwanda, Rwanda
Corresponding author email: dsdawoud@gmail.com
[3]PhD, Senior Lecturer, Faculty of Applied Sciences, National University of Rwanda.

ABSTRACT

This paper describes the current state-of-the-art of side channel attacks (side channel cryptanalysis) on embedded systems. Major countermeasures proposed in the literature are reviewed and the challenges faced by embedded system designers while considering the implementation of such countermeasures are highlighted. The large number of problems to be taken into account and the highly application-dependent character of side channel attacks make it impossible to advise adequate countermeasure as general solutions. The designer must therefore start by defining the adversary the device must resist against and with the resources available for him choose appropriate countermeasures against this adversary. The paper gives an overview of an ongoing research work which aims at finding novel hardware-based techniques to implement countermeasures for timing and power analysis attacks.

Keywords: Electromagnetic Analysis Attacks, Embedded Systems, Power Analysis Attacks, Side-Channel Attacks, Timing Analysis Attacks

1. INTRODUCTION

Security has been the subject of intensive research in the context of general-purpose computing and communications systems. However, security is often misconstrued by embedded system designers as just adding security features, such as specific cryptographic algorithms and security protocols, to the system. Such narrow view of embedded system's security assumes that the attackers will use traditional mathematical cryptanalysis to break the system security. Embedded system designers should instead be aware that the cryptographic primitives they add to guarantee the security of the device are implemented in a program that will run on a given processor and in a given environment which is normally insecure. The insecure environment makes attacks possible especially if the attacker has access to the embedded device and can play around with it. Such types of attacks are called "implementation attacks". One of the forms of such attacks is known as "side-channel attacks" (SCA) and represents real threats for embedded systems. 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. This fact adds a new dimension to the problem of securing the embedded system: not only the software functionality running on this system should be proven secured but also the hardware implementation as well. In reality, implementing security in embedded systems is more challenging than doing it for general purpose processors. This is due to the limitation of resources available on the resource constrained embedded devices, such as processing capabilities, memory space limitation, and battery limitation. Time-to-market constraint can also be a pressuring factor for designers developing secure embedded systems. Field Programmable Gate Arrays (FPGA's) can help to achieve a short design time as they allow many design iterations with short design cycles. However, Side channel attacks have been identified as an important threat against cryptographic applications implemented on FPGA's (Boneh, *et al,* 1997). The designer using an FPGA should keep this fact in mind in order to guarantee security of the system. In fact security

is not a non-functional requirement but it is a new dimension that designers should consider throughout the design process, along with other metrics such as cost, performance, and power. The designer must think of all possible software and hardware countermeasures against potential SCA. However, security may be in contradiction with other design metrics (for example the countermeasures of timing attacks vs. performance) and this needs a form of optimization. This represents new challenges for embedded systems designers. Implementing security in embedded systems require new approaches for all aspects of embedded system design from architecture exploration to system implementation. Security processing, which refers to the computations that must be performed in a system for the purpose of security, can easily overwhelm the computational capabilities of the embedded system. This is known as the "security processing gap". The next challenge is the "assurance gap," which relates to the gap between functional security measures (e.g., security protocols) and their actual secure implementations. Beside these gaps there is another feature for embedded systems that adds more challenges: flexibility. An embedded system is often required to execute multiple and diverse security protocols and standards in order to support (i) multiple security objectives (e.g., secure communications), (ii) interoperability in different environments, and (iii) secure processing in different layers of network protocol stacks. Furthermore, with security protocols being constantly targeted by hackers, it is not surprising that they keep continuously evolving. It is, therefore, desirable to implement security in a flexible (programmable) way that easily adapt to changing requirements. Section 2 of this paper introduces the different types of "implementation attacks". Side channel attacks are considered in Section 3 where a new idea of hardware-based countermeasure implementation is presented. Section 4 includes a conclusion of this paper.

2. IMPLEMENTATION ATTACKS

Implementation attacks that target a cryptographic device include active attacks, passive attacks and/or combinations of both types of attacks.

2.1 Passive Attacks

Such attacks use the cryptographic device in its intended environment and can obtain cryptographic keys by physical leakage. Passive attacks simply observe the device's behaviour during its processing without disturbing it. The information flow used to perform such attacks can be the power consumption of the device, electromagnetic radiation, timing information on the cryptographic service, or error messages obtained. Passive attacks can be classified into two kinds: **Side-Channel Attacks (SCA)** and **Logical Attacks.** SCA are discussed with more details in Section 3. Logical Attacks use external logical functions of the cryptographic device and look for specific software or protocol bugs that can be exploited. Direct access to the cryptographic device is not necessary.

2.2 Active Attacks

This kind of attacks ranges from changing the environmental conditions to the physical opening of the cryptographic device. Active attacks try to interfere with the device proper functioning; for example, fault-induction attacks will try to induce errors in the computation. Active attacks can be: (i) **Non-invasive Attacks** which only exploits externally available information.. The information can be, for example, running time information or power consumption leaked by the device. (ii) **Semi-invasive Attacks** which requires opening the cryptographic device for specific types of attacks. These attacks require de-packaging of the chip to get access to the chip surface, but do not tamper with the passivation layer - they do not require electrical contact to the metal surface.

3. MAIN FORMS OF SIDE-CHANNEL ATTACKS

3.1 Probing attacks

One way to attack an embedded device is to de-package it and observe its behaviour by branching wires to the data bus or observing memory cells with a microscope. To make observation easier,

the attacker may try to slow down the clock provided to the chip, so that successive states are easily observable. An introduction on probing attacks can be found in (Anderson, 2001), and a very good overview of ways to de-package a smart card and probe its content is given in (Kämmerling & Kuhn, 1999).Countermeasures for probing attacks include for some embedded systems, e.g. smart cards, adding a passivation layer, which is basically a shield covering the chip, in order to prevent from observing its behaviour.. Some embedded systems are equipped with detectors, for example in the form of additional metallization layers that form a sensor mesh above the actual circuit and that do not carry any critical signals. All paths of this mesh are continuously monitored for interruptions and short-circuits, and the device has to refuse processing and destroy sensitive data when an alarm occurs. Another technique consists of monitoring clock frequency and refusing to operate under abnormally low (or high) frequency.

3.2 Fault induction attacks

When an electronic device stops working correctly, the most natural reaction is to get rid of it. This apparently insignificant habit may have deep impact in cryptography, where faulty computations are sometimes the easiest way to discover a secret key. A recent and powerful cryptanalysis technique consists in tampering with a device in order to have it perform some erroneous operations, hoping that the result of that erroneous behaviour will leak information about the secret parameters involved. Fault attacks may be classified into: (i) **Simple Fault Induction Attack** which exploits a direct relationship between a faulty result and the secret key in the implementation. The most prominent key exposure attack is published on an RSA-CRT implementation that can succeed with one fault. (ii) ***Differential Fault Induction Attack*** that needs a certain number of faulty computational results using the same cryptographic key. Here one assumes that faults are caused in a transient way. The faulty outcomes are used to reduce the key space. Fault induction attack on RSA with Chinese Remaindering Theorem (CRT) (Joye, et al, 1999) is probably the best example for showing how easy to deploy and how effective such types of attack are. Countermeasures for fault induction attacks are relatively easy: the cryptographic device itself shall check that the result obtained is correct. In the simplest way, this can be done by computing the same operation twice. For crypto-based invertible operations, the result can be checked by calculating the inverse operation internally, e.g. checks that the cryptogram gives the message back. Other countermeasures as in case of the RSA-CRT make use of certain control variables that are checked regularly. For a hardware-based countermeasure, Sergei P. Skorobogatov and Ross J. Anderson (Skorobogatov & Anderson, 2001) proposed self-timed dual-rail-logic circuits that include an alarm mechanism.

3.3 Timing attacks

Usually the run time of a program is merely considered as a constraint, a parameter that must be reduced as much as possible by the programmer. More surprising is the fact that the run time of a cryptographic device can also constitute an information channel, providing the attacker with invaluable information on the secret parameters involved. This is the idea of timing attack which was first introduced by Kocher (Kocher, 1996), and then practically used against an RSA implementation based on the Montgomery algorithm (Dhem, et al, 1998). Many other works were devoted to improvements of this kind of attacks. In its final form, it was possible to recover a 512-bit key using between 5 000 and 10 000 timing measurements. In (Handschuh, 1999), Handschuh presents a timing attack against RC5, which requires 220 measurements to succeed. The same author propose a timing attack against the AES (Rijndael), recovering a key with 4 000 measurements. Timing attack can also be used in conjunction with other side-channel attacks. For example, Walter and Thompson (Walter & Thompson, 2001) recently proposed very efficient attacks based on the analysis of time variations in RSA sub-operations. Their method is based on the ability to observe partial timings, namely those of each individual multiplication of the exponentiation loop. This information cannot be obtained by sole time measurements, no matter how precise they are, but several other side channels (e.g. SPA) can be used for this purpose.

Countermeasures for timing attacks are relatively easy to implement: it is generally sufficient to make sure that the execution time is data-independent. Countermeasures can be of two types: hiding variations or blinding.

Hiding variations - The simplest way to hide variation is to make the computation strictly constant-time. This, however, would contradict the performance constraint. It would imply a very severe performance drawback, especially for asymmetric cryptosystems, since this constant time would of course be that of the slowest possible case. Another way to achieve countermeasure for schemes that exploit the reduction of multiplication algorithms consists of modifying the multiplication algorithm (normally it is Montgomery algorithm) so that an additional subtraction is always carried out, even if its result is simply discarded afterwards. Dhem (Dhem, 1998) proposed an improvement of these multiplications schemes, allowing several modular multiplications to be chained with only one extra reduction being performed after the last multiplication. Similar methods were proposed by Hachez et al. (Hachez & Quisquater, 2000). They studied conditions under which a square and multiply algorithm can be carried out without the need for additional reductions between Montgomery multiplications. Three main attacks against RC5 and AES can easily be defeated, respectively by ensuring that the shift instruction's run time is not dependent on the number of positions to shift, and by ensuring that the time *x* operation will be implemented in a constant-time way (see (Schindler, *et al*, 2001) for more details).

Hiding internal state - The second type of countermeasure consists in hiding internal state, so that the attacker cannot simulate internal computations anymore (see (Kocher, 1996)). Once again, this countermeasure is only guaranteed to defeat a well known attack, and we cannot rule out the possibility of a completely new timing attack, making blinding ineffective. In fact, some attacks, combining timing and power analysis, have actually been proposed.

Timing Attack against Code-based Cryptosystems - Besides the classic number theory encryption, there is another alternative based on coding theory. In 1978, R. J. McEliece presented this type of encryption for the first time. This work became the reference for all the researches in public key cryptography based on coding theory (McEliece, 1978). The security of McEliece has been studied by many authors and they proposed good solutions to the main problem of the original proposal, namely the huge size of the key. Some of the results led to a public key of 6500 bits (Niederreiter, 1986) and 4096 bits (Li, *et al,* 1994). There is a very few number of publications which dealt with SCA attacks on code-based encryption. The authors of (Strenzke, et al, 2008) pointed out the possibility of using time side-channel attacks on code-based cryptography. In (Shoufan*, et al*, 2009) the authors studied this possibility more closely. They devised an appropriate countermeasure that is implemented in the decoding algorithm. The author of (Strenzke, 2010) introduced a timing attack against the secret permutation used in generating the key. The countermeasures in case of code-based systems is called reactive; at specific points during computation a check for a certain condition has to be performed. In the case when this condition is fulfilled some values have to be modified before the computation is continued.

3.4. Power analysis attacks

Power consumption of a cryptographic device may provide much information about the operations that take place and the parameters involved too. This is the idea of simple and differential power analysis, first introduced by Kocher et al. in (Kocher, *et al*, 1999). Simple Power Analysis (SPA) is a technique that involves directly interpreting power consumption measurements collected during cryptographic operations. Differential Power Analysis (DPA) (Agrawal, *et al,* 1999) is of great concern as it is very effective in finding the secret key and can be mounted quickly with off-the-shelf devices. The attack is based on the fact that logic operations have power characteristics that depend on the input data. It relies on statistical analysis

to retrieve the information from the power consumption that is correlated to the secret key. DPA have been identified as an important security issue for cryptographic applications on FPGA's (Akkar, *et al*, 2000), (Anderson & Kuhn, 1996). It is expected that general countermeasures, which have been developed in the past on the architecture level or the algorithmic level, can also be implemented on an FPGA. To some extent, they will help in concealing the supply current variations. Yet none has ever proven really successful and effective in thwarting the DPA. Considering the fact that the power consumption of a single logic gate is controlled by both the logic value and the sequence of its input signals forms the basis of DPA, it is our opinion that the solution of the DPA attacks is on the logic level. Simple Power Analysis (SPA) is a technique that involves directly interpreting power consumption measurements collected during cryptographic operations. SPA can reveal the sequence of instructions executed; accordingly it can be used to break cryptographic implementations in which the execution path depends on the data being processed.

Power attacks against code-based cryptosystems

Attacks on McEliece Public Key Cryptography (PKC) - The author of (Strenzke, 2010) showed that public key cryptography based on McEliece suffers also from high risk of leaking side information through side-channels. He also presented a possible power attack against the key generation phase when using McEliece protocol. During the key generation phase, certain operations involve the same secret value repeatedly. Countermeasures must be considered during implementation of the key generation to avoid breaking the key using some sophisticated power attacks.

Power attacks against Stern identification scheme - Stern identification and signature scheme is one of the applications using code-base cryptography. It is an interactive zero-knowledge protocol which aims at enabling a prover P to identify himself to a verifier V. In (Li, *et al,* 1994), the authors described the first implementation of Stern protocol on smart card. They used a key with length 694 to guarantee the security of the information on the card with acceptable hardware. To secure the Stern scheme against SPA and first order DPA, the authors showed that there are four parts of the protocol dealing with sensitive data: Matrix-vector product, Hash function, Permutation method, and Pseudo-random generator. They also described how to efficiently hide the leakage of information using random masks. Applying first order masking to Stern's Protocol induces only a very small timing overhead and an acceptable memory overhead, since almost all the performed operations are linear. Our research group works on finding hardware countermeasures to power and/or timing attacks. Power attacks countermeasures should include implementations techniques which tend to minimize, or ideally, avoid the variation in power dissipation. The reader can find in literature some proposals that solve the problem at gate level. They propose the use of equivalent AND, OR, NOR gate circuits that consume the same amount of power irrespective to the inputs. Our approach consists of finding one logic circuit that can work in any mode (AND, OR, NOR, etc.) using some control signals. The main condition is that the circuit will consume the same power irrespective to the input signals. In other words, our target is to end up with a form of homogenous array of circuits that consume fixed (approximately) amount of power. The second level of our research is on the processor level; i.e. how to design a system of normal processors with fixed power consumption. Our preliminary researches showed that this approach can solve the problem of timing attacks. However, the two approaches result in increasing the hardware resources. On of the aim of our research work is to find a way of minimizing such increase.

3.5 Electromagnetic analysis attacks

Any movement of electric charges is accompanied by an electromagnetic field. The current going through a processor can characterize it according to its spectral signature. *Electromagnetic*

attacks, first introduced by Quisquater and Samyde (Quisquater & Samyde, 2000), and further developed in (Gandolffi, *et al,* 2001) exploit this side channel by placing coils in the neighbourhood of the chip and studying the measured electromagnetic field. The information measured can be analyzed in the same way as power consumption (simple and differential electromagnetic analysis - SEMA and DEMA), but may also provide much more information and are therefore very useful. Agrawal et al. (Agrawal, *et al,* 1999) showed that EM emanations consist of a multiplicity of signals, each leaking some different information about the underlying computation. As far as software countermeasures are concerned, electromagnetic attacks and power attacks (and, to a lesser extent, timing attacks) are, in many respects, very similar: the way the side channel leaks information differs, but the type of leaking information is roughly the same. Software countermeasures do not try to reduce the signal amplitude, but rather to make the information it conveys useless by obscuring the internal parameters. To prevent SPA or SEMA, one has to make the execution flow as constant as possible, or, in other words, independent of the manipulated data. Several authors (Akkar, et al, 2000), (Borst, 2001) have shown that most of the power consumption (85%) is due to the instruction executed, compared to the data involved (10%). In case of using algorithms that contain multiply and square instructions, it is probably good practice to use the same function to implement the two instructions (Gomulkiewicz & Kutylowski, 2002). For hardware-based countermeasures the information available in scientific literature is pretty limited. This is either because of embedded devices manufacturers are not very keen to giving details about the countermeasures they develop for their products or because such countermeasures are patented and protected. Most of the authors discuss basic hardware countermeasures in a very general way. Generally speaking, it is also important to make sure that a countermeasure against one specific attack does not make another one easier.

4. CONCLUSION

The side-channel attacks (SCA) on embedded systems are getting a great importance among researchers and industry. This comes from the fact that it is possible to obtain encryption keys using side-channel attacks. These kinds of attached are based on timing, power and electromagnetic analysis of data leaked by the cryptographic embedded system. The solutions of such attacks start during the design phase of the system; it is one of the main functional requirements. Countermeasures can be implemented in both software and/or hardware, and then the designers must consider it during the design phase. Our research group is working on finding new design techniques that can be used to countermeasure power and/or timing attacks. A very brief idea about our line of research is given.

5. REFERENCES

Agrawal, D., B. Archambeault, J.R. Rao, and P. Rohatgi, 1999, *The EM side channel*, Lectures Notes in Computer Science (LNCS), vol. 1717, Springer-Verlag.

Akkar, M.-L., R. Bevan, P. Dischamp, and D. Moyart, 2000, *Power analysis, what is now possible*, Advances in Cryptology - ASIACRYPT '00 (T. Okamoto, ed.), Lectures Notes in Computer Science (LNCS), vol. 1976, Springer-Verlag.

Anderson R. and M. Kuhn, 2001 *Tamper resistance-a cautionary note*, Proc. of the second USENIX workshop on electronic commerce (Oakland, California), pp. 1-11.

Boneh, D., R.A. DeMillo, and R.J. Lipton, 1997, *On the importance of checking cryptographic protocols for faults*, Advances in Cryptology – EUROCRYPT '97, Konstanz, Germany (W. Fumy, ed.), LNCS, vol. 1233, Springer, 1997, pp. 37-51.

Borst, J., 2001, *Block ciphers: Design, analysis and side-channel analysis*, Ph.D. thesis, K.U.Leuven.

Dhem J.F., 1998, *Design of an efficient public-key cryptographic library for risk-based smart cards*, Ph.D. thesis, Universite catholique de Louvain – UCL Crypto Group - Laboratoire de microelectronique (DICE).

Gomulkiewicz M. and M. Kutylowski, 2002, *Hamming weight attacks on cryptographic hardware - breaking masking defences*, Computer Security - ES-ORICS 2002(D. Gollmann, G. Karjoth, and M. Waidner, eds.), Lectures Notes in Computer Science (LNCS),vol. 2502, Springer-Verlag.

Gandolffi K., C. Mourtel, and F. Olivier, 2001, *Electromagnetic analysis: Concrete results*, Proc. of Cryptographic Hardware and Embedded Systems (CHES 2001), Lecture Notes in Computer Science, vol. 2162, Springer, pp. 251-261.

Hachez G. and J. Quisquater, 2000, *Montgomery exponentiation with no final subtraction: Improved results*, vol. 1965, Springer-Verlag, pp. 293-301.

Handschuh, H., 1999, *Cryptanalyse et securite des algorithmes ce secrµete*, Ph.D. thesis, Ecole Normale Superieure des Telecommunications.

Joye, M., A. K. Lenstra, and J. Quisquater, 1999, *Chinese remaindering based cryptosystems in the presence of faults*, Journal of cryptology 12, no. 4, 241-245.

Kämmerling O. and M. G. Kuhn, 1999, *Design principles for tamper-resistant smartcard processors*, Proc. of USENIX Workshop on Smartcard Technology (Smartcard '99).

Kocher, P., 1996, *Timing attacks on implementations of Diffe-Hellman, RSA, DSS, and other systems*, Advances in Cryptology - CRYPTO '96, Santa Barbara, California (N. Koblitz, ed.), LNCS, vol. 1109, Springer, pp. 104-113.

Kocher, P., Jaffe J., and B. Jub, 1999, *Differential power analysis*, Proc. of Advances in Cryptology -CRYPTO '99 (M. Wiener, ed.), LNCS, vol. 1666, Springer-Verlag, pp. 388-397.

Li, Y. X., R. H. Deng and X.-M. Wang, 1994, *On the equivalence of McEliece's and Niederreiter's public-key cryptosystems*, IEEE Transactions on Information Theory, volume 40, number 1, pages 271-273

Messerges, Th., 2000, *Securing AES finalists against power analysis attacks*, Fast Software Encryption: 7th International Workshop - FSE '00 (B. Schneier, ed.), Lectures Notes in Computer Science (LNCS), vol. 1978, Springer-Verlag.

McEliece, R. J., 1978, A *Public-Key System Based on Algebraic Coding Theory*, Jet Propulsion Lab, DSN Progress Report 44, pages 114-116

Niederreiter, H., 1986, *Knapsack-type cryptosystems and algebraic coding theory*, Problems Control Inform. Theory, Vol.15, number 2, pages 159-166

Quisquater J. and D. Samyde, 2000, *A new tool for non-intrusive analysis of smart cards based on electro-magnetic emissions: the SEMA and DEMA methods*, Eurocrypt rump session.

Schindler W. Q, and F. Koeune, 2001, *Improving divide and conquer attacks against cryptosystems by better error detection correction strategies*, Proc. of 8th IMA International Conference on Cryptography and Coding, pp. 245-267.

Shoufan A., Strenzke F., Molter H. G., and Stöttinger M., 2009, *A Timing Attack Against Patterson Algorithm in the McEliece PKC*, in ICISC 2009

Strenzke F., 2010 *A Timing Attack against the secret Permutation in the McEliece PKC*, preprint 2010

Strenzke F., E. Tews, H. G. Molter, R. Overbeck and A. Shoufan, 2008, *Side Channels in the McEliece PKC*, The Second International Workshop on Post-Quantum Cryptography PQCRYPTO, Lecture Notes in Computer Science, Vol. 5299.

Skorobogatov S. P. and R. J. Anderson, 2001, *Security engineering*, Wiley & Sons, New York.

Walter C. D., and S. Thompson, 2001, *Distinguishing exponent digits by observing modular subtractions*, Proc. of RSA conference 2001, 2001.