High Security Smartcards

Organisers: M. Renaudin[1], F. Bouesse[1]
Presenters: Ph. Proust[2], J.P. Tual[3], L. Sourgen[4], F. Germain[5]

[1]Tima Laboratory, 38031 Grenoble, France - Marc.Renaudin@imag.fr
[2]Gemplus Corporate R&D Security Technologies, 13705 La Ciotat, France - Philippe.PROUST@gemplus.com
[3]Axalto - Schlumberger, 78431 Louveciennes, France - Jean-Pierre.Tual@Louveciennes.sema.slb.com
[4]STMicroelectronics Smartcard division, 13106 Rousset, France - laurent.sourgen@st.com
[5] Central Directorate of Information Systems Security, 75007 Paris, France - fabien.germain@sgdn.pm.gouv.fr

Abstract

New consumer appliances such as PDA, Set Top Box, GSM/UMTS terminals enable an easy access to the internet and strongly contribute to the development of e-commerce and m-commerce services. Tens of billion payments are made using cards today, and this is expected to grow in a near future. Smartcard platforms will enable operators and service providers to design and deploy new e- and m-commerce services. This development can only be achieved if a high level of security is guaranteed for the transactions and the customer's information.
In this context, smartcard design is very challenging in order to provide the flexibility and the powerfulness required by the applications and services, while at the same time guaranteeing the security of the transactions and the customer's privacy. The goal of the session is to introduce this context and highlights the main challenges the smartcard designers/manufacturers have to face.

I) Introduction

Smartcard's architecture is first introduced to point out the tremendous amount of hardware and software technologies involved in its design. A smartcard is a complex embedded system taking advantage of:
- state of the art silicon technologies,
- secure high performance and low power micro-processors, sometimes assisted by dedicated coprocessor units to improve speed and/or security,
- several types of memory, non volatile memories such as ROM, EEPROM, Flash and fast volatile memories (RAM),
- communication interfaces which can be contactless,
- analog parts and sensors to protect the chip against attacks,
- embedded software, based on secure operating systems, virtual machines, firewalls, cryptography and other specific applications…

Smartcards are subject to many different types of attacks aiming at tampering the chip or parts of it in order to retrieve secret information [1]. Cryptography and cryptanalysis are both rapidly evolving at the theoretical and at the implementation, both software and hardware, levels. This knowledge is essential to evaluate the risk associated to the attacks and implement appropriate countermeasures.

State of the art non-invasive and invasive attacks [2] are reviewed in the first part of the session. The next two parts explain the challenges hardware and software designers are facing to design secure smartcards. Even though presented in two separate parts it is pointed out that software and hardware components of a smartcard must be co-designed to achieve a high level of security. The last section introduces the standardization process and the so-called common criteria, together with the governmental and certification bodies.

I) Threats and tampering means

This section introduces all recent powerful cryptanalysis methods which are considered as real threats for smart card security applications. Cryptanalysis is defined as an art or a science of analyzing and recovering secret information hidden in cryptography systems [3]. Its different fields of action are represented by the tree depicted in figure 1. Each branch of the tree stands for categories, subcategories and types of attacks according to the knowledge discipline they involve. All types of methods use the following classes of attacks: Chosen clear message attacks, Known clear message attacks, Known

1530-1591/04 $20.00 (c) 2004 IEEE

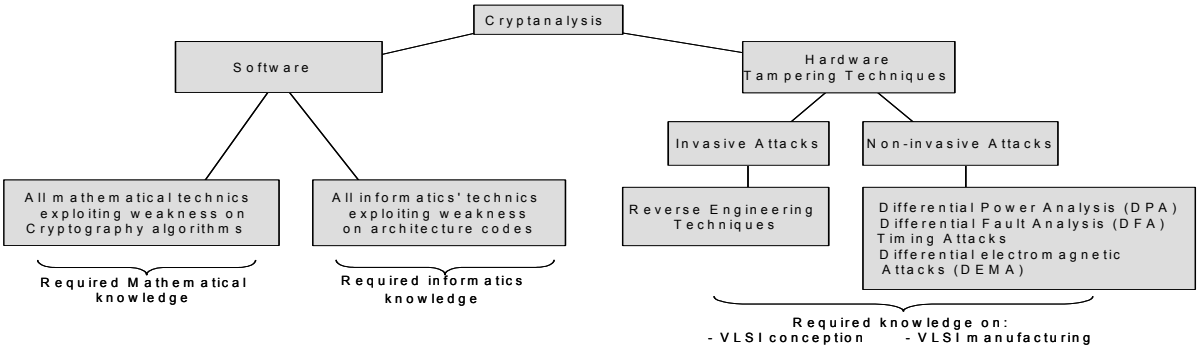

Figure1: Cryptanalysis: Different fields of action

clear message pattern attacks and no known pattern of bits in the clear text messages [3].

1) Hardware Attacks

Cryptanalysis techniques focused on chip's Hardware represent real menace to smart card security since new effectiveness attacks were discovered. We distinguish between two majors' subcategories.

- Invasive Attacks

Invasive attacks are based on reverse engineering to access the chip surface directly and microprobing techniques to observe, manipulate and interface with the integrated circuit [4]. These attacks require special laboratory equipments and destroy packaging in the process.

- *Non-invasive Attacks*

This new type of attacks called side-channel attacks exploit hardware implementation weaknesses of cryptographic algorithms. Paul Kocher has specified methods [5] to break device secret key by measuring and analyzing device power consumption. Several side-channel attacks have been characterized and implemented:

- Fault attacks: such attacks occur in abnormal environment conditions to generate malfunction in devices that create a window of vulnerabilities. Glitch attacks on power or clock signal may provide access to a chip's secure area [2]. Moreover, the Differential Fault Analysis (DFA) exploits computational errors in cryptosystem using algebraic properties of modular arithmetic to find keys [6].

- Timing attacks: measuring and analyzing the amount of time required for running non constant cryptographic algorithm may provide information about data processed. This technique takes advantage of the direct dependence between computation time and data processed [7].

- Power Analysis: Analyzing the electrical activity of the power supplies and of the interface signals of the devices during normal operations enables hackers to retrieve sensitive information. Single and Differential Power analysis are well known for their efficiency [5] [8].

- Electromagnetic Attacks: Instead of using the current to perform SPA or DPA, this technic is exploiting the electromagnetic emissions of the chips. This type of attacks called SEMA (Single Electromagnetic Attack) and DEMA (Differential Electromagnetic Attack), appear to be more powerful than current based attacks [9-10].

2) Software Attacks

Software attacks are divided in two subcategories: the first subcategory groups all attacks focused on mathematical methods which allow breaking the cryptographic algorithms [2]. This type of attack is not addressed in this hot topic. The second subcategory groups all attacks exploiting vulnerability programs embedded in smartcard. These attacks are well known for breaking software systems. Buffer overflow and Trajan Horse may respectively inject deliberately useful code and open a window of vulnerabilities in software system.

II) Design for security in smart card software

From the security standpoint, Smart-Card embedded SW is basically in charge of interfacing with the Hardware, managing the logical security within the card, controlling the risk management policy of the issuer, implementing main functional countermeasures against SW threats. Versus a pure functional design, SW security adds-on may represent 30 to 50% additional embedded code. This is a strong constraint, given the small HW resources available in a smart card chip. The design of secure SW for smart cards must comply with three fundamental requirements:

1) Clearly identify the assets to protect

Such assets include customer specific data (such as Secret Keys, PIN code and related counters), Life Phase Locks, Data structure parameters (size, flags, access rights,…), balance and currency (which is of course a must for e-purses), random numbers. More generally, in open smart-card architectures (e.g. JavaCards), applets and more generally any piece of ROM code can be such asset.

2) Clearly identify the types of attacks the SW must be able to protect (in a sole or combined way SW and HW).

Software attacks aim at taking profit from bugs in embedded Operating Systems or Virtual Machine specification, design or implementation. They can also originate from poor cryptography implementation of personalization errors.

At the difference with Hardware attacks, they are often independent from the underlying hardware and can often be performed with very light equipments. Such examples of attacks include timing attacks or brute force attacks against cryptographic algorithms.

It would nevertheless be misleading to imagine that secure SW design is a completely disconnected activity from the secure HW part. In practice, the most dangerous attacks are combined HW+SW attacks, taking profit from insufficient cooperation between HW and SW layers. Such attacks are often put in practice after hackers have identified that the Chip has some intrinsic vulnerabilities. Such examples of combined attacks are Observation attacks (DPA, SPA) or Fault injection attacks (DFA). Knowledge of security characteristics of underlying HW (at both architecture and security levels) is always a pre-requisite for « good » embedded SW design.

3) What is the insurance level the embedded SW must reach ?

As the target applications of smart cards are often very sensitive from both business or citizen perspective (e.g. debit/credit, Pay-TV, Health, e-purse), getting proofs of security of the final product (HW + embedded SW) by independent evaluators is becoming a standard requirement from most card issuers today. With the generalization of the Common Criteria methodology, and in a similar way to HW, the embedded SW development process becomes more and more formal in all its major steps (TSP and FSP development, SW test, configuration management) and more and more supported by advanced CAD tools (example: formal methods for security checking at high specification levels, co-design approach for hardware and software co-development, formal proof inserted in the design flow for data integrity checking between different levels of SW description, intellectual property definition for SW [e.g. third party Java applets],…)

The three major requirements described before, translate in some major smart-card SW design guidelines:

1) Security guided SW development principles

Good design principles include considerations such as layered/firewalled SW architecture, application isolation, RAM and EEPROM data protection and more generally logically access to application data only under OS control (which may restrict the use of technologies such as JavaCard or embedded VMs)

2) Tight integration with HW

This is specially true for everything that concerns the HW/SW interface, cryptographic computation, language acceleration, use of Memory Management schemes (MMU, ACL,…), use of complex HW execution mechanisms (cache, pipe-line)

3) Use a set of well documented countermeasure mechanisms, providing the best compromise between cost and performance for a given security result

Based now on more than 20 years of experience, such mechanisms (specific re-usable modules) belong to well defined classes such as:

- RAM management
  - Access rights control
- Card and/or application Life Cycle
  - Card interrupt
- EEPROM management
  - Programme Flow management
- Data Structure
  - Counters
- Cryptography
  - Security supervision

Each of those mechanisms can be implemented with some given “strengths” and induces a given design cost (dedicated buffer, additional writing in EEPROM, CRC computation, additional cycles needed,…).

It is the art of smart-card SW designer to take all these constraints into account for developing embedded

code able to achieve the design security requirements expressed before.

III) Design for security in smartcard hardware

The hardware design for secure products (like smartcard) has to satisfy 3 main requirements:

- The design must include specific functions for security (like random generator, specific local encryption engines….)
- All functions involved or related to security must be implemented in a robust way (minimise leakage and sensitivity to disturbs).
- The functions and their implementation need to be proven for security.

The first point is mainly a matter of architecture definition, and should include appropriate experience in the analysis phase. Once defined, they could be implemented like any other function, assuming than proper criteria for robustness are defined. One must evaluate the nature of the threats to assess the risk to be analysed.

The second point is the most critical. All functions handling critical data or instructions can leak confidential information (direct current analysis like SPA, or more sophisticated differential approach like DPA or High order DPA). They are also sensitive to external perturbation (like supply glitches, light or other radiations impulses….) that may create fault in the execution of an operation. If not handled properly, faults are exploited by DFA based attacks. The key difficulty is to evaluate these aspects at design level, before making silicon while most of these problems are mainly measured on real circuits.

To answer such request special simulation had to be developed (like counting and analysis logic transition during gate level simulation to anticipate supply current correlation). The need for accurate post layout simulation with all parasitic elements is a major difficulty. It takes a lot of computing power to extract the data, to simulate the possible current correlation, and the feedback loop to improve the circuit is long and complex (no automatic tools so far to do so). This method may be used for evaluating leakage, and examples of circuit optimised for DPA at design level exist now, although without standardize methods or tools (mainly in house or in university flows). The savings are nevertheless enormous (no need for silicon iteration, no delay on market…). The resistance to fault injection is more difficult to evaluate, as it is very difficult to analyse all fault consequence (junction leakage, uncontrolled switches….). Various techniques can then be applied, like detection of possible fault cause (like glitch sensor, light sensors…) but are limited as new fault sources appear frequently.

Another approach is to assume that fault may occur but detect their effect (in secure product you don't need to keep functionality under fault injection, but being able to detect the fault or fault effect before it can be used – Reset of the circuit is then good enough). This calls for careful design (no undefined states, redundant encoding scheme, parity control….). In that case the final verification can only be done on real silicon.

The last point focusses on the proof that the expected level of security is achieved. All techniques for verification (formal or semi formal description, property checking…) have to be used to demonstrate (in the mathematical way if possible) that the design will behave as specified. Also the design have to be deeply characterized, so that behaviour that may compromise security in some aspects are well known and documented so that security countermeasures can be defined and applied at system level (software…). Usage of more formal methods, standard security evaluation schemes (like common criteria...) are essential to raise the level of confidence and the risk assessment capability of new designs.

IV) Evaluation and certification

When someone says that a system is secured he faces a challenge: people must trust in him and people must trust in the system. But which assurance is given ? Behind security there is confidence. To achieve confidence people need a reference, this is the role of the certification.

France developped a national scheme for the process of evaluation and certification. It was created by a Prime Minister's text and by a decret. The French evaluation and certification scheme will be described in order to understand the process to trust in the security offered by a system.

Nevertheless, this result is national and the need is an international result. The common criteria were created by a group of nations in order to have an international recognition of the security. In this presentation, the common criteria will be presented with a brief history and their different components. We will focuse on the development aspect to highlight the needs for the design process and the design tools to increase the security.

V) Conclusion

The paper introduces the context and gives another view of the challenges smartcards designers and manufacturers are facing. It is clear that the design of a smartcard, integrating many different kinds of hardware blocks (analog digital), as well as multi-layers softwares, is in itself complex. The security criterion is adding another dimension which requires specific skills, experience, and of course to adopt adapted design strategies and methodologies able to take into account the whole system avoiding security holes. In this domain, the design methodologies are evolving towards

more formalization and verification. Modelling and formalizing the threats and attacks is essential to specify the corresponding countermeasures. Then, design for security requires tools able to take security requirements into account both for hardware and software developments. Finally, it is very important to be able to prove, as much as possible, that the design is reaching a given level of security and this, before fabrication. The goal is to reduce the design cycle of secure systems and to precisely control the security / cost tradeoff.

References

[1] Ross Anderson, Security Engineering, Published by John Wiley & Sons, Inc. 2001 ISBN 0-471-38922-6.

[2] R. Anderson and M. Kuhn, "Tamper Resistance-a cautionary Note" , The Second USENIX Workshop on Electronic Commerce Proceedings, November 1996, pp. 1-11.

[3] Bruce Schneier, Applied Cryptography, John Wiley & Sons, 1996 ISBN ISBN 0-471-12845-7

[4] D. Samyde, S. Skorobogatov, R. Anderson and J. J. Quisquater , On a New Way to Read Data from Memory, First International IEEE Security in Storage Workshop, December 11 - 11, 2002, Greenbelt, Maryland.

[5] P. Kocher. J. Jaffe and B. Jun. Differential Power Analysis: Leaking Secrets. Advanced in Cryptology – Proceeding of Crypto '99, Springer Verlag, LNCS 1666, pages 388-397. One version of the paper is available online at http://www.cryptography.com/dpa/technical.index.html.

[6] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In 17th Annual International Cryptology Conference (CRYPTO), volume 1294. Springer-Verlag, August 1997.

[7] P. Kocher. Timing attacks on implementations of Diffie-Hellman, RAS, DSS, and other systems. In N. Koblitz, editor, Advances in Cryptology - CRYPTO '96, Santa Barbara, California, volume 1109 of LNCS, pages 104-113. Spinger, 1996.

[8] T. S. Messerges, Using Second-Order Power Analysis to Attack DPA Resistant Software, Proceedings of Cryptographic Hardware and Embedded Systems (CHES 2000), 2000, pp. 238-251.

[9] D. Agrawal, B. Archambeault, JR. Rao, and P Rohatgi. The EM side-channel(s), In Cryptographic Hardware and Embedded Systems Conference (CHES '02), 2002.

[10] K. Gandolfi and C. Mourtel and F. Olivier. Electromagnetic Analysis: Concrete Results. In Workshop on Cryptographic Hardware and Embedded Systems (CHES), volume 2162. Springer-Verlag, May 2001.