Overview about Attacks on Smart Cards

by Wolfgang Rankl, Munich

The following article is a condensed version of the chapter about smart card security in the Smart Card Handbook from Wolfgang Rankl und Wolfgang Effing which will be published in the 3rd edition at John Wiley and Sons in September 2003.

20. February 2003

The possibility of protecting and securing data is one of the main advantages of cards with electronic chips (smart cards) in comparison to all other data carriers, such as cards with magnetic strips or diskettes. Consequently, the chip hardware must be designed in an optimum fashion to meet this purpose; this includes the corresponding cryptographic procedures for securing the secret data. However, security is not only dependent on the specialised hardware of the microcontroller or on the cryptographic algorithms implemented in the operating system software. The security regarding the application of smart cards and the design principles applied by the developers to meet that purpose are of fundamental importance. The essential property of a smart card is its ability to offer a secure environment for data and programmes.

1 Attacks and Countermeasures during the Development

As early as in the development phase of the microcontroller hardware and the operating system software of the smart card, a variety of security measures is taken. Similar to the quality of a product security must be considered from the very start of the development of a product because it cannot be added to the design of a product at a later point of time.

1.1 The Development of the Smart Card Microcontroller

The development of the hardware of a smart card microcontroller takes many months, and it is carried out by a few persons at a manufacturer of semiconductors in secured and monitored facilities. The corresponding computer systems for the design of the semiconductor are typically connected to an independent network that has no point of contact to the rest of the world. This guarantees that modifications to the design of the chip cannot be implemented and the internal architecture of the chip cannot be determined by external sources.

Comprehensive inside-knowledge is required to manipulate the chip design in a way that security is adversely affected. Therefore, such an attack is rather unlikely. In addition, independent test institutes evaluate the architecture and the security measures of almost all smart card semiconductors nowadays.

Protection: Design Criteria
There are some fundamental criteria concerning the design of the functions of a smart card microcontroller. On the one hand, the measures against static and dynamic attacks have to be effective. Sensors and protective elements are of little use if they can be worked around easily or if they possibly do not take effect. One such example would be sensors on a semiconductor chip that require such a large amount of space that makes it possible to destroy them with a needle, thus they would not be effective anymore.

One very important design criterion which is different from the standard components is the requirement that there must not be any undocumented mechanisms or functions under any circumstances (*"that's not a bug, that's a feature"*). Most of the time such undocumented "features" are not completely tested since a few persons only know them and therefore they often have a number of flaws or weak spots. As they are not documented, they might not be included in the evaluation of the hardware and they might possibly be used for attacks at a later point of time. Consequently, the use of such undocumented features is strictly prohibited, even if they might be of substantial help to the developers.

Protection: Unambiguous Chip Number
During the development of the semiconductor all hardware-related security elements must be designed and implemented for the microcontroller to be produced. Apart from the sensors and protective layers, a WORM memory (*write once, read multiple)* is used. During the production of the semiconductor, an unambiguous chip number is stored in that memory. The chip is thus individualised and can be clearly traced, and the smart card produced can be identified unambiguously throughout the system. In addition, this number can be used for deriving keys offering the option of setting up no-go lists that make it possible to withdraw suspicious smart cards from the circulation.

It should be noted, however, that although this number cannot be changed on the original chip, there is no protection from copies of this chip being made by a freely programmable microcontroller. Therefore, security mechanisms must not be based on a certain chip having a certain number stored in its WORM memory. This unambiguous number can only be the basis for real cryptographic security mechanisms.

1.2 The Development of the Smart Card Operating System

The development of software for smart cards is implemented analogous to current software development principles. Certain general conditions have to be met regardless of the development methodologies (waterfall model, spiral model, etc.) that are used.

It is essential that the computers used for the development of such software are incorporated in their own and completely independent network that does not allow any access from outside. The development tools, such as compiler and chip simulators, are software packages whose functionality has been examined in independent tests. In some cases, even two different compilers are used in order to ensure the correctness of the result. As a rule, the use of software whose origin cannot be determined exactly is prohibited, as this might be a possible way to manipulate the development tool in order to change the programme code to be created.

Protection: Development Principles
Similar to the hardware development process, undocumented features must not be implemented in the course of software development. It would be very helpful, for instance, to include commands allowing any desired memory area to be read in order to convert the time-consuming black box tests typical of smart cards into white box tests. If, however, one of these commands were to be forgotten in the programme, the secret keys could be determined from the real smart cards. In order to avoid such an attack approach from the very beginning, including any dump commands in the programme is unwanted even if this would help to reduce cost intensive development time. Nevertheless, deadline pressure and the constantly increasing complexity of smart card operating systems have led to the undermining of this principle. In order to guarantee that these commands accompanying the development process will not be included in real smart cards under any circumstances,

special tests are carried out during the manufacturing process of smart cards to ensure the non-existence of such commands.

Another principle calls for the development of a programme never to be carried out by one single programmer. This is ruled out by the software quality assurance on the one hand, but on the other hand, security issues regarding attacks also require a dual control principle to be applied at any time during software development. This is a very effective method of making inside attacks harder to be accomplished since at least two developers have to agree on the development process at any time. In addition, internal source code inspections are carried out on a regular basis to ensure the quality and to monitor the development process.

Once the software development process has been concluded, it is common that the complete source code developed as well as its functionality are examined by independent test institutes within the scope of a software evaluation. The major reason for these time and cost intensive examinations is the exclusion of software faults, however, they also make it impossible for a developer, for instance, to hide a Trojan horse in the operating system.

Protection: Distribution of Knowledge

If several persons are working on one task, the result is much more robust against attacks due to different experiences and opinions of the persons involved. The principal of the distribution of knowledge (*shared secret*) counteracts the approach of "everybody knows everything about anything." On principle, during the development of security components, the complete knowledge should not be concentrated on one person, as this person becomes open to attacks. Comparable to some military areas, knowledge gained during the development is distributed among various groups of persons, which makes it possible for experts to have discussions on one topic while there is nobody who knows everything.

The same holds true for the completion of the smart card operating system, i.e. loading tables, programme code and configuration data into the EEPROM. Apart from higher flexibility, this also constitutes a security aspect. From that point on the complete knowledge on the operating system lies with the chip manufacturer, who receives the completely assembled ROM programme code for manufacturing the masks. Those parts of the operating system that are stored in the EEPROM are not available to the chip manufacturer, which makes it impossible for him to gain knowledge on the complete security mechanisms and the functionality of the operating system if he analyses the ROM code.

2 Attacks and Countermeasures during the Manufacturing Process

Attacks during the manufacturing process of chips or smart cards are typical insider attacks as the corresponding manufacturing environments are closed environments. The access is closely regulated and each access is recorded. Nevertheless, the manufacturing process must be included in the security aspects because some technically very interesting and effective attacks can be carried out here.

Protection: Authentication in the Manufacturing Steps

Even during the production of wafers, the smart card microcontrollers are individualised by a chip number and protected with a transport code. Regarding the latest operating systems, each chip has an individual transport code, and authentication is mandatory for each production-related access to the chip. While this makes the production more time-consuming and requires a security module in the corresponding machines, it significantly increases security.

An obvious attack during the manufacturing process is the infiltration with dummy chips or dummy smart cards that behave identical to the regular components, however, they have, for instance, a command for dumping the memory. Of course, the replacement of a

real chip with a dummy chip is only possible once the wafers have been divided into single dice. This type of attack can be demonstrated by means of a smart card for digital signatures: during initialisation, the attacker replaces a real smart card with a dummy card. This card is then initialised with real data and personalised afterwards.

As this smart card has all the functions of a real smart card, the keys for the asymmetrical cryptoalgorithm in the microcontroller would also be generated. The required data can be obtained from the initialisation and personalisation data. In the following, the attacker would have to manage to regain access to his smart card and he could use his special dump command to read the secret signature key from the card. As the trust centre has signed the corresponding public key, it has been verified as being valid. Thus, the attacker has now all the information needed to produce an unlimited number of duplicates of such a validated card.

Such an attack is unrealistic because the organisational features at the manufacturers make it impossible for chips or smart cards to be brought into or taken from the corresponding premises. In addition, the authentication between the smart card and the security module of the manufacturing machine required for all manufacturing steps prevents the replacement of chips or cards.

3 Attacks and Countermeasures during Card Use

In contrast to previous phases of the card's lifecycle, the access to the component to be attacked - the smart card - usually requires far less effort from attackers once the smart card has been issued. This is one of the reasons why the probability of an attack is relatively high especially during the phase in which the card is used.

The following chart lists and describes examples of attacks that can almost be called classical attacks. The descriptions of the attacks represent the state-of-the-art, and they are intended for persons who are inexperienced in the topic of smart card security. They provide a competent overview so that already known critical mechanisms are not be used again due to a lack of knowledge. The countermeasures specified are available for the defence of such attacks. These, in turn, can be worked around by modified attack scenarios, which leads to the well-known cat and mouse play of measures and countermeasures at attack and defence.

The scenarios portrayed do not serve as manual for cracking the security of a smart card system since all of them are known and publicly available [Koemmerling 99]. They do not constitute serious threats for the security of today's modern smart cards because the attacks specified have already been considered by the corresponding protective measures. A few years ago, however, those attacks might still have been successful.

Figure 1 Overview of typical attacks that had an influence on systems equipped with smart cards, sorted by discovery date. In the following text, the specified attacks as well as the corresponding first countermeasures are described in more detail.

Known since	Attack	Short Description of the Attack
before 1990	Bugging of data transmission	The data transmission between the terminal and the card can be bugged by attaching wires to the module. The countermeasure was the introduction of Secure Messaging.
≈ 1990	Dissolving the passivation	Dissolving the passivation-layer around the microcontroller serves as prerequisite for gaining physical access to components on the microcontroller. The countermeasure was the in-

		troduction of passivation detectors on the microcontrollers.
≈ 1990	Manipulation of data transmission	The data transmission between the terminal and the card can be manipulated at will by electrically insulating the contact fields of the module and the wires attached to the module. The countermeasure was the introduction of Secure Messaging.
≈ 1991	Deletion of the EEPROM by using UV light	By deleting the EEPROM with the help of UV light, for instance, counters can be reset to their original values. The countermeasure was the introduction of light sensors on the microcontrollers.
≈ 1991	Setting up memory card equivalents	Both the functionality of the memory card and the secret authenticity feature can be emulated by setting up equivalents of memory cards. The countermeasure was the introduction of challenge response authentication on the memory cards.
≈ 1992	Disconnecting the voltage supply	The retry counter of the PIN can be avoided by disconnecting the voltage supply during the PIN verification process. The countermeasure was the prophylactical increase of the retry counter prior to the PIN verification process.
≈ 1993	Stop clock frequency	Conclusions can be drawn regarding the RAM content by halting the clock frequency and analysing the RAM with the help of electron beam testers. The countermeasure was the introduction of low-frequency detectors on the microcontrollers.
≈ 1993	Manipulation of the microcontroller by means of laser cutters	The components on the microcontroller can be manipulated with the help of laser cutters. The countermeasure was the introduction of protective layers around the microcontrollers.
1995	Timing attack	Due to a lack of knowledge, a dependency between the key and the validity was created during the implementation of many cryptoalgorithms. This can be used for determining secret keys. The countermeasure was the realisation of noise-free cryptoalgorithms.
≈ 1995	Bugging the bus with microprobe needles	The buses on the microcontroller can be bugged with microprobe needles. The countermeasure was scrambling the buses on the microcontrollers.
1996	DFA	Secret keys of cryptoalgorithms can be calculated by selectively introducing miscalculations to the processor. The countermeasure was the introduction of glitch detectors on the microcontrollers as well as the corresponding precautionary measures in the cryptoalgorithms.
≈ 1996	Manipulation of the microcontroller by means of a FIB	The components on microcontrollers can be manipulated with the help of a FIB. The countermeasure was the introduction of protective layers around the microcontrollers.
1997	Exhaustive key search in DES	High-performance computers and computer networks can calculate DES keys within a few hours by means of a brute-force-attack. The countermeasure was the use of triple DES.
1997	Statistical distribution of PINs	When the four-digit PINs were generated in the German EC-card system, there was no statistical rectangular distribution; therefore, some PIN values occurred much more frequently than others. The countermeasure was the use of an improved generating algorithm.
1998	SPA/DPA	The data processed can be determined by the power consumption of the processor. The countermeasures were: the introduction of random waiting periods to the processor, the use of processors with steady power consumption as well as a number of precautionary measures within the software of the microcontroller.
1998	COMP 128	Due to a design flaw of the authentication algorithm COMP 128 used by some network providers, it is possible to determine the secret keys by means of a brute-force-attack. The countermeasure was the use of other authentication algorithms and the limitation of the number of authentications.
1998	Processor disruption	The processor can be disrupted at critical stages while processing the machine code by attacking the processor (e.g. by means

	of light flashes). The countermeasures were the corresponding detectors on the microcontrollers as well as a large number of precautionary measures in the software.

3.1 Attacks on a Physical Level

Manipulations in the area of semiconductors require a large amount of technical effort. Depending on the attack scenario, this may include microscopes, laser cutters, micromanipulators, focused ion beams, equipment for chemical removal procedures and fast computers for the analysis, the recording and evaluation of electrical procedures on the chip. This equipment and the corresponding knowledge required for their application are only available to very few specialists or organisations, which drastically reduces the probability of an attack on a physical level. Nevertheless, a manufacturer of cards and/or semiconductors must always assume that a potential attacker can use all the equipment required for such an attack; therefore, the corresponding security features must be included in the hardware.

The following paragraphs explain the most important protective mechanisms most frequently used in smart card microcontrollers.

3.1.1 Static Analyses of the Smart Card Microcontroller

Protection: Semiconductor Technology
The chip's structures (width of conductor path, size of transistors, etc.) have reached the limits of what is technically possible today. The typical width of the structure ranges between 0.35 µm and 0.13 µm, which itself is no longer a special technological feature. The transistor density on the silicon, however, has reached the top limit of what can currently be achieved by using the typical lithographical manufacturing procedures. Only these very fine structures make it difficult to obtain information from the chip with the help of analytical procedures. Therefore, semiconductor technologies with structural sizes of one micrometer and below are currently considered to be secure. In future, this will certainly change.

Protection: Chip Design
So-called standard cells, which, for instance, contain a processor core or certain types of memory, are often used for the design of semiconductor-related components. The advantage lies in the fact that a manufacturer of semiconductors can use these standard elements to quickly produce a variety of different high-quality chips. This procedure was developed for the mass production of products not taking into account security aspects. Hence, it is not used for the production of smart card microcontrollers due to the following reasons: the structure and the mode of functioning of standard cells is publicly known, which would provide potential attackers with too much information and make their work much easier.

Protection: Buses on the Chip
All internal buses on the chip connecting the processor with the three different types of memory, ROM, EEPROM and RAM, do not lead to the outside and therefore cannot be contacted. There is no possibility for an attacker to bug or influence the address, data, or control bus of the microcontroller in order to gain knowledge of the memory contents. Usually, the buses are integrated into the lower layers of the semiconductor, which makes it difficult to contact them directly from the surface. In addition, the buses on the chip are scrambled either statically, individually per chip or individually per session, so that the

function of the individual bus circuits cannot be determined from external sources. There are smart card microcontrollers with the feature of constantly changing the scrambling process of the buses even during a running session.

Protection: Memory Design

ROM is the memory medium for most programmes. The content of a ROM generally used for industrial purposes can be read with a light microscope bit by bit. Consequently, it is not very difficult to combine the bits to form bytes and to combine the bytes to form the complete ROM code. In order to avoid this analysis, the ROM is not integrated into the uppermost layers that can easily be accessed but into the lower silicon layers. This prevents an optical analysis to be carried out.

If, however, the chip's front side is pasted on a carrier, and the chip is ground off from the backside, the contents of the ROM could be obtained. In order to avoid this, the smart card microcontrollers are exclusively equipped with ion implanted ROMs, whose data contents are not visible, neither in visual nor in IR or UV spectrum. This also protects the chip to a large extent from so-called selective etching. This procedure is used to etch the semiconductor in such a way that the contents of the ROM become optically visible.

Protection: Protective Layers (*Shield*)

One danger lies in the analysis of electrical potentials on the chip during operation. Provided that the scanning frequency is high enough, this makes it possible to measure charge potentials, i.e. voltages, on very small areas of the crystal, and thus conclusions can be drawn on the data contents of the RAM during operation. This can very reliably be avoided by adding conducting metallic layers to the corresponding memory area or to the whole chip. If these metallic layers are removed chemically, the chip will no longer be functioning, as these layers are required for supplying electric voltage needed for the correct functioning of the chip. In many cases, various protective layers are arranged on top of each other, and they are permanently checked on intactness.

In addition, semiconductor technology allows the implementation of meander shaped current-carrying structures on the complete surface of the chip or on areas that are at high risk (e.g. low-frequency detectors). They can easily be monitored via measuring the resistance or the capacity or they can be implemented into the function of the chip so that the chip will be switched off immediately if these structures are damaged. The security can be increased further, if the linkage of the meander shaped structures is modified during one session. This prevents the meander from being bypassed with the help of a FIB (*Focused Ion Beam*).

Protection: Scrambling the Memory

Similar to the scrambling of buses that has been implemented a long time ago, scrambling of memories is increasingly used on microcontroller chips. The security is based on keeping the scrambling pattern of the memory cells secret. Memory can be scrambled easily and little additional space is required on the chip. Without the corresponding scrambling information it is extremely difficult for an attacker to determine the way the memory cells are addressed.

Protection: Memory Encryption

Apart from the swapping of data in the memory (*scrambling*), modern smart card microcontrollers also offer the possibility of encrypting the memory and even a part of the registry of the processor on a batch or chip individual level. During this process, the corresponding data is encrypted and decrypted in real time during reading and writing. In addition to the key, some chip types offer the option to include the memory address in the en-/decryption process. This results in equal data having different values at different positions in the memory after the encryption process. Individual keys for each session can especially be used for RAM areas.

If an attack was successful and data could be read from the memory, the secret key would still be necessary to gather information that makes sense. This significantly increases the effort required from the attacker since he either must know at which position this key is stored or he must read all the data available on the chip.

3.1.2 Dynamic Analyses of the Smart Card Microcontroller

Protection: Monitoring the Passivation Layer
After the microcontroller has been produced on the silicon, a passivation layer is added, which prevents oxidation, for example by atmospheric oxygen, as well as other chemical processes from taking place on the surface of the chip. First, this passivation layer must always be removed in order to manipulate a chip. It should be taken into consideration, however, that the passivation layer can be removed chemically, but the chip is subject to a high oxidation risk, which can destroy it rather quickly. A sensor circuit can determine via resistance or capacity measuring whether this passivation layer is still present. If it is no longer present or if it is damaged, either an interruption in the chip's software can be triggered or the complete chip will be disconnected from the hardware, which reliably prevents all dynamic analyses.

Protection: Voltage Control
Each smart card microcontroller is equipped with a voltage control system. This system is responsible for switching off the component in a controlled manner when the upper or lower limits of the operating voltage are exceeded. This secures the software in such a way that an operation in the limit ranges at which the chip is no longer fully functional is impossible. If no voltage control were in place, an operation in these limit ranges could e.g. lead to the programme counter of the processor not to run stable anymore, which for instance leads to uncontrolled leaps within the programme or which causes miscalculations in the processor. This abnormal behaviour can be used as a starting point for determining secret keys by means of the differential fault analysis (DFA) specified in the text below.

Especially voltage control is of high significance for the security of the microcontroller. One possible attack would be to destroy the corresponding detectors useless, e.g. by means of a FIB (*Focused Ion Beam*), and to start the actual attack afterwards. This is the reason why in many cases the components required for the security of the microcontroller are protected especially well, so that a manipulation will be recognised and the smart card will be deactivated automatically.

Protection: Frequency Monitoring
Generally the core rate of the smart card is regulated externally, thus the internal clock speed is determined from the outside. This offers the possibility - at least theoretically - of running the microcontroller in single-step operation. This would lead to excellent analysis possibilities especially regarding the measuring of the power consumption (*Power Analysis*) and the determination of electrical potentials on the chip. In order to avoid this type of attack, a functional group of components for the detection of sub or over-frequencies is integrated on the chip. It prevents the defined core rate from being lowered inadmissibly.

In order to protect the microcontroller from the danger of being operated in single-step mode it is sensible to secure the low-frequency detectors with protective layers, so that the microcontroller cannot be manipulated unnoticed.

Protection: Scrambling of the Buses
Many smart card microcontrollers scramble the buses addressing the memory, which are only accessible internally on the chip. This means that the individual bus circuits are not

arranged in an ascending or descending order but in a mixed-up and several times swapped order next to each other or even isolated by layers on top of each other. This is an additional obstacle for potential attackers, as they do not know which bus circuit has which function or address.

Originally this scrambling of the bus circuits was only introduced in a static variant, i.e. with identical scrambling on each chip. Thus, in the medium term it would not be a real problem for an attacker to determine how the circuits are scrambled and to consider this accordingly in a bugging campaign.

There is, however, an improvement of the security by introducing scrambling processes of the buses that individual for each chip. This chip-individual scrambling is not achieved by producing different exposure masks for the buses of each chip as this cannot be realised technically at the moment and it would be far too expensive. The scrambling is carried out by scramblers, which are directly located on the memory and can be controlled by the individual chip numbers. This procedure can be effected rather effortless with semiconductors and it makes a bugging campaign much more difficult. A scrambling process that is individual for each chip and each session can be realised as well by using variable input values in the scramblers.

Dynamic Analysis and Defence: Measuring the Power Consumption of the CPU

In June 1998 Paul Kocher, Joshua Jaffe and Benjamin Jun published a document on simple power analysis (SPA) and differential power analysis (DPA) [Kocher98a].

The principle of the simple power analysis (SPA) is rather simple. An analogue-digital converter is used to measure the power consumption of a microcontroller by determining the drop in voltage at a resistor connected in series at a high temporal resolution. Due to the relatively simple structure of the CPUs of smart card microcontrollers the internal processes and the processed data lead to measurable and interpretable effects on the power consumption. To make it clearer, one can imagine that the same programme sequence with the same data leads to a certain cycle of the power consumption of the processor. If this programme is run with different data, the cycle of the power consumption differs. This deviation is used to determine the processed data.

In comparison to the simple power analysis (SPA) the differential power analysis (DPA) makes it possible to discover even smaller differences in the power consumption of the microcontroller. For this purpose the power consumption is first determined during the processing of known data and then during the processing of unknown data. The measuring is usually repeated various times and the mean value is calculated to eliminate the noise. After the measuring is terminated, the difference is determined and from the result the unknown data can be concluded.

The power analysis of smart card microcontrollers is an attack to be taken very seriously for unprepared hardware and software. The reason for this is that in some microcontrollers there can well be dependencies of the power consumption from the corresponding machine instruction and also from the data processed with this machine instruction. Furthermore, the required effort for a successful attack regarding the measuring equipment is rather small. There are, however, a number of effective countermeasures, which are based on improved hardware on the one hand and on modified software on the other hand.

The simplest hardware-related solution is the installation of a fast voltage regulator on the chip that ensures power consumption independent from the machine instruction and data with the help of a shunt resistor. The implementation of random noise sources on the chip is also an effective solution. A technically more challenging solution is the use of a modified semiconductor design of the processor that leads to constant power consumption. However, some of these attempts increase the power consumption of the microcontrollers, which is undesirable in certain areas of application such as telecommunication. A simple countermeasure during an SPA/DPA critical process can also be the activation of components not required for this process such as CRC checksum generator or numerical co-

processor with random data as input values in order to produce an artificial noise of the power consumption.

The use of random wait states in the processor significantly complicates the synchronisation process during the power analysis without having the disadvantage of higher power consumption. A similar solution is used for smart card microcontrollers with an on-chip core rate generation by permanently varying the core frequency randomised within defined limits.

Regarding the software-related countermeasures there is a tremendous range of solution variants by now. A few exemplary solutions are specified shortly in this section. The easiest approach is the exclusive use of machine instructions with very similar power consumption. Machine instructions with a significant deviation from the average power consumption must not be used in the assembler code anymore. A further attempt is the introduction of different orders for the same calculations of cryptoalgorithms that are chosen randomly. This makes it much harder for the observer to recognise a convergence between known and unknown machine instructions and processed data. A similar attempt is the use of chip-individual tables for the S-boxes of the (triple) DES algorithm.

In order to complicate the data collection required prior to a successful power analysis, all keys should be secured by irreversible retry counters. Moreover, it is necessary to block the free access to all commands of the type INTERNAL AUTHENTICATE in which any data can be sent through a cryptoalgorithm of the smart card. By limiting the use of commands in such a fashion the collection of reference data for the power analysis at a later point is prevented.

Analysis and Defence:
Measuring the Electromagnetic Radiation of the CPU

At least theoretically conclusions can be drawn from the electromagnetic radiation on the internal sequence of events on the smart card microcontroller, which is a similar process as the differentiated power analysis. SQUIDs (*superconducting quantum interference devices*) can be used to measure magnetic fields of low extension and strength. The evaluation can be carried out analogous to SPA/DPA. A first attempt of such an attack is cited by Karine Gandolfi [Gandolfi 01]. However, the technical effort required is high and the necessary knowledge of the internal structures of the semiconductor is not generally available. In addition, semiconductor components can be protected very effectively from these types of attacks by arranging various conducting paths on top of each other so that a magnetic field can be determined with sensitive detectors but not the conduction that carries the current.

3.2 Attacks on a Logical Level

Attacks on the security of a smart card on a logical level require above all an understanding of the communication and the flow of information between the terminal and the smart card. It is not so important to understand the processes of the hardware level but to know the processes of the software. From an information-technological point of view the exemplary scenarios described here are considered to be one level higher than the attacks making use of the hardware properties.

Attack and Defence: Dummy Smart Card

The most conceivable attack is the use of a smart card that has been self-programmed and enhanced with a number of analysis and protocol functions. A few years ago this could hardly be carried out because a few companies could only purchase smart cards and the corresponding microcontrollers. By now, however, smart cards and configuration pro-

grammes can be purchased from various companies on the free market. This increases of course the number of possibilities available to an attacker. But irrespective of that with some effort and skills a functioning smart card can be build from a small plastic plate and a standard microcontroller in an SMD casing. At least a card that functions like a real smart card regarding the electric properties during data transmission. Today, this type of smart card can be purchased from a vast number of traders via the Internet. New possibilities regarding smart cards are also offered by Java technology with the help of which own programmes can be created easily and loaded onto a dummy card.

Such a dummy card can be used to record a part of the communication with the terminal, which can be evaluated later. After several attempts it may then be possible to execute a part of the communication process just as with a real smart card.

It can be doubted, however, that a real advantage can be obtained from this as all professionally designed applications have a cryptographic protection for important actions. As long as one does not know the secret key the attack is over at the latest when it comes to the authentication. This attack would only be successful if the attackers know the secret key or if the complete application is running without cryptographic protection. If such an application exists it is rather improbable that the advantage gained from this attack is of such a high significance that it justifies the great effort.

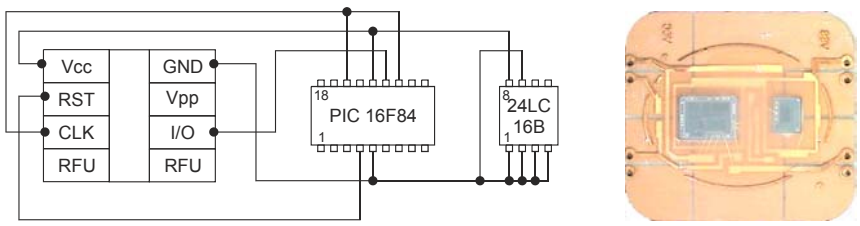

Figure 2 Typical substitute circuit for smart card microcontroller built from discrete standard components (microcontroller PIC 16F84, EEPROM memory chip 24LC16B). The components fit into a typical smart card module so that a distinction from a real smart card microcontroller is impossible without examination of the module. This circuit and variants of it can be found on numerous pages in the Internet.

Attack: Bugging of the Data Transmission

In order to bug and, when required, manipulate the data during a session a slightly modified smart card is used. An electrically insulated dummy contact is attached to the I/O interface. The original I/O interface is then no longer electrically connected to it. The newly created (dummy) contact and the original I/O contact are connected to a fast computer. Depending on the programming this computer can cut out or insert any data during the communication between the terminal and the smart card. If the computer is fast enough, neither the terminal nor the smart card will be able to determine any difference from the regular data transmission during the manipulated communication.

It is understood that this method can drastically influence the sequence of a session. The question whether there is any benefit for the attacker primarily depends on the application used on the smart card. An approved design criterion determines that the security must not be impaired by bugging, cutting out or inserting data during the communication. If this criterion is not observed an attacker will certainly gain a benefit this way.

Attack and Defence: Cutting off Power Supply

One attack that was still very successful a few years ago in many smart cards is the cutting off of the power supply at a certain point of time while a command is executed. The background of this attack is the fact that in case of conventional programming all write operations on EEPROM pages are executed one after the other. If the programmer of the command has arranged the order of the write operations in an unwise fashion, the attacker can gain an advantage by cutting off the power supply at the right time.

However, the designers of operating systems do know an effective countermeasure: atomic orders. They have the property of being atomic, i.e. indivisible. This means that they are either carried out completely or not at all, which is an adequate protection against the attack mentioned above.

Attack and Defence: Power Analysis at PIN Comparison

By combining physical measuring of a parameter and varying logical values a technically very interesting attack on comparison features such as the PIN can be carried out. It concerns all mechanisms where data is sent to the smart card and compared with the stored value while a retry counter is increased depending on the result of the comparison.

This is based on the principle of measuring the current of the smart card, which can be carried out, for example, via the drop of voltage at a resistor connected to the Vcc circuit. If the corresponding command is sent to the card together with the comparison data, it is possible to determine via power measuring prior to receiving the return code whether the retry counter was increased or not. If, in case of a positive comparison result, the return code is sent out before the retry counter was increased, the comparison value could be determined. For this purpose, the comparison value is sent to the smart card in all its variants and the card is always switched off in a negative case before the retry counter is increased. The positive case can be clearly identified by the corresponding return code that is sent before the retry counter is increased.

There are two basic methods to avoid this attack: The easiest way is to increase the retry counter prior to each comparison and to decrease it again, when required. Now, it does not matter when the attacker interrupts the power supply, he can never gain an advantage as the retry counter has already been increased. The second variant requires more effort but fulfils the same protective function. After the comparison the retry counter is increased in a negative case and written into an unused EEPROM cell in a positive case. Both write accesses take place at the same time so that the attacker cannot draw any conclusions from the comparison. He is not informed about the result of the comparison until he receives the return code. At that time it is too late to prevent the write access to the retry counter by disconnecting the power supply.

Attack and Defence: Time Analysis at PIN Comparison

Programmers are always concerned that programs run as fast as possible. Generally, this is an important feature. This feature of run-time optimisation can also be used for a promising attack, though. If a smart card is in the process of verifying the PIN, the corresponding comparison routine carries out a byte-by-byte comparison of the entered PIN and the stored PIN. A programmer who does not pay attention to security will program the routine in such a way that a difference in the comparison of the two PIN-codes leads to an immediate abort of the routine. This will result in minimal run-time differences that can be measured with suitable equipment (e.g. memory oscilloscope). These differences can be used by an attacker to determine the secret PIN in a relatively simple way.

Only a few years ago the attack mentioned above was still successful. In the meantime, though, this type of attack is known and the comparison routines are designed in such a way that as a matter of principle always all digits of a PIN are compared. Thus, there is no time difference between positive and negative comparison results.

Protection: Noise-free Cryptoalgorithm

Even in the early 90's some cryptoalgorithms were still used with major differences in run-time depending on the key and the uncoded text. With the key space being reduced that way the attacker can use a brute-force attack to search for the secret key. How long the search takes depends to a large extent on the noise of the algorithm. The larger the time

differences the smaller the key space, and the simpler and faster the key search. If the exact implementation of the corresponding cryptoalgorithm on the target computer is known, this can also be used as an additional reference for creating timetables. This type of attack - a timing attack - was published in 1995 in a publication from Paul Kocher [Kocher 95], dealing especially with time dependencies at RSA and DSS.

Principally a timing analysis is very dangerous for the security of a smart card. But as it has been known for a long period of time, all of today's smart cards exclusively use noise-free cryptoalgorithms, i.e. the time for encrypting and decrypting is independent from the input values. As a result this type of attack was defeated.

An additional security feature in some applications is the extra retry counter of all authentication keys so that only a certain number of unsuccessful authentications can be carried out. Once the retry counter has reached its maximum value, the smart card is locked against any further authentication attempts.

Manipulation: Differential Fault Analysis *(DFA)*

In 1996 Dan Boneh, Richard DeMillo and Richard Lipton published a paper [Boneh 96] describing a theoretical model how secret keys of asymmetrical crypto-algorithms can be calculated by causing hardware faults.

Only two months later Eli Biham and Adi Shamir published an extension of the Bellcore attack with the name differential fault analysis (DFA) [Biham 96] which now also included symmetrical crypto-algorithms such as the DES algorithm. As a result many smart cards were open to a new attack method to be taken seriously - at least theoretically.

The basic principle of both attacks is rather simple: In the first step any given uncoded text is encrypted with the key to be decoded and the encrypted text is kept. After that the smart card is interrupted during the processing of the cryptographic algorithm externally, for instance, by ionised or high frequency rays so that one individual key bit is modified at any position. The result is a key text that has not been encrypted correctly due to the corrupted bit. This process is repeated a number of times and the results are kept for the analysis. The remaining part of determining the key is pure mathematics and is specified comprehensively in the publications mentioned above.

The power of the attack lies in the fact that it is not even necessary to know at which position of the secret key a bit was corrupted. Biham and Shamir quote in their publication that in the case of one corrupted key bit, 200 key text blocks are enough to generate the complete secret DES key. If a real Triple DES (168 bit) is used instead of the DES, the number of required key texts does not increase significantly. Even if more than one bit is changed, this attack is still effective. Only the number of wrongly encrypted key texts required is increased.

In practice, this type of attack is not so simple as it seems. If possible, only one bit or at least very few bits are supposed to be modified. If the complete microcontroller is subject to high-frequency microwaves, so many bits are modified that the processor typically crashes irretrievably. Therefore, the attackers try to get the CPU to make one single wrong calculation with the help of e.g. intentionally created glitches[1] in the power supply or core-rate generation. If the filters situated on the corresponding input lines cannot neutralise such a glitch, the desired miscalculation of the processor can occur.

A smart card, however, is not without protection against the Bellcore attack or a DFA, if corresponding care has been applied beforehand. The simplest defence is to calculate the crypto-algorithm in the smart card twice and to compare the two results. If the results are identical, then no attempt was made to corrupt any bits from the outside. In this context, it is assumed that an intentional implementation of faults can never modify the same bits in the smart card. This is an assumption that is very close to reality, because if a targeted modification of certain bits should ever be possible in a smart card processor, then there are much simpler and quicker attacks than a DFA.

[1] Glitches are very short losses or increases in voltage

The big disadvantage of a dual-calculation is the additional time required, which may cause problems. This especially concerns attacks on time-consuming asymmetrical crypto-procedures such as RSA or DSS. A further effective defensive measure against differentiated fault analyses can be achieved by always exclusively encrypting different uncoded texts. The simplest solution is the use of a random number in front of the uncoded text to be encrypted. Consequently, the crypto-algorithm always encrypts different data and a DFA is no longer possible.

What the Bellcore attacks and the differentiated fault analysis all come down to is the fact that they are indeed dangerous attacks that can be successful in case of insufficiently equipped smart cards. However, within a short time period after the two attack methods became known, all smart card operating systems and applications were secured to that respect. Therefore, nowadays neither Bellcore attacks nor DFA constitute a serious danger anymore.

Attack and Protection: Disrupting the Processor

Similar to the use of the differentiated fault analysis (DFA) when attacking secret keys of crypto-algorithms, it can be attempted to disrupt the processor in order to influence the sequences in the programme code. An attack that has been known as light attack to the manufacturers of smart cards and smart card microcontrollers and to some system houses since 1998 was published in mid-2002 by Sergei Skorobogatov and Ross Anderson [Skorobogatov 02] as *Optical Fault Induction Attacks*. The publication describes an arrangement in which a regular flashlight is flanged to the camera adapter of a conventional light microscope. This is used to flash a very limited area of the RAM of a standard microcontroller (PIC16F84). This arrangement makes it possible to selectively set certain bits in the RAM of this microcontroller to the value 0 or 1 provided that it has no protection against this type of attacks.

In order to disrupt the processor, glitches on the supply lines, light flashes on the chip or on parts of the chip or even high-frequency can be applied [Lamla 00]. If at the correct point of time the jamming of the programme sequence is triggered, a query, for instance, can be influenced selectively. Figure 3 shows a simple example. The programme function specified has the task of transmitting the content of a send buffer whose limits are defined by a starting address and an ending address. If the attacker manages to interfere selectively with the query for the end of the send buffer, the data following the send buffer will also be transmitted to the terminal. If this memory area of the RAM were to include the secret key of a crypto-algorithm, then this key could be obtained without permission using this method.

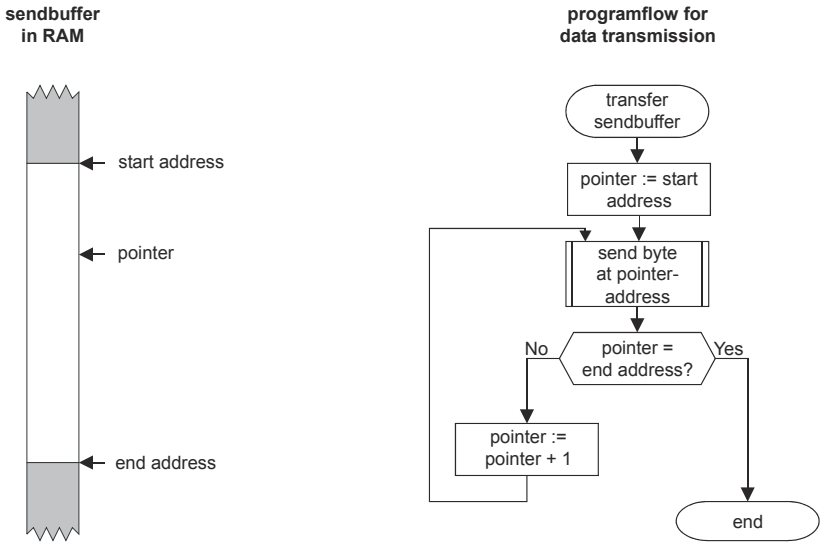

Figure 3 Example of a non-robust programme sequence for the transmission of data from a send buffer which can be attacked successfully by disrupting the processor.

The defence against this attack consists of various steps. It is important that the smart card microcontroller is equipped with the corresponding sensors to detect all disruption attempts of the processor. This can be voltage sensors detecting glitches, and a large number of corresponding light sensors on the chip.

The second protective layer must be realised within the software. The programme code quoted in the example can be made much more robust by substituting the "=" query with a "<=" query. As an additional countermeasure, the query can be carried out twice, where the timeframe between the two queries should be randomly chosen. As a result, the attacker would have to use two light flashes for manipulating the query and, moreover, would have the problem that he cannot exactly predict the point of time for the second light flash.

In addition, all confidential data should be deleted from the RAM immediately after their use or it should be temporarily encrypted. In order to further reduce the effects of this attack, it is also sensible to encrypt all secrets (e.g. PIN, key) in the EEPROM. That is to say if an attacker should manage to read parts of the EEPROM by manipulating queries, he would only receive encrypted data as a result, which are useless to him. If a MMU is available, it can also be configured in such a way as to monitor whether certain limits are kept when data is sent. Furthermore, modern processors can detect illegal machine code and invalid addresses and react correspondingly. This defence scenario provides a good impression of how a serious attack can be blocked off by the corresponding cooperation of protective measures of hardware and software.

4 Conclusion

Of course, it is virtually impossible to design a complete system or even one single smart card in a perfectly secure way that cannot be breached by anything or anybody. In the end, one only has to use a high enough effort for an attack in order to be able to infiltrate or manipulate any system. However, each potential attacker, be it consciously or subconsciously, will always make some sort of benefit analysis for himself and for his targets. After all, the result he will receive by breaching a system must be worth the work, the money and the time he is putting into the operation. If the result - be it monetary value or reputation among experts and the world - does not justify the effort, nobody will put too much energy into breaching a system or a smart card. This is one of the main criteria for designing a secure system with smart cards.

5 Appendix

[Biham 96] Eli Biham, Adi Shamir: A new cryptoanalytic attack on DES, Internet, 1996

[Boneh 96] Dan Boneh, Richard A. DeMillo, Richard J. Lipton: On the Importance of Checking Computations, Internet, 1996

[Gandolfi 01] Karine Gandolfi, Christophe Mourtel, Francis Oliver: Electromagnetic Analysis: Concrete Results, Workshop CHES 2001, 2001

[Kocher 95] Paul C. Kocher: Timing Attacks on Implementations of Diffie-Hellmann, RSA, DSS, and Other Systems, Internet, 1995

[Kocher 98 a] Paul C. Kocher, Joshua Jaffe, Benjamin Jun: Introduction to Differential Power Analysis and Related Attacks, Internet, 1998

[Kömmerling 99] Oliver Kömmerling, Markus G. Kuhn, Design Principles for Tamper-Resistant Smartcard Processors, USENIX Workshop on Smartcard Technology, Chicago USA, 10–11 Mai 1999

[Lamla 00] Michael Lamla: Hardware Attacks on Smart Cards - Overview, Eurosmart Security Conference, Marseille, 13–15 Juni 2000

[Skorobogatov 02] Sergei Skorobogatov, Ross Anderson: Optical Fault Induction Attacks, Internet, Mai 2002