Fault attacks on RSA with CRT: Concrete Results and Practical Countermeasures*

C. Aumüller** P. Bier*** W. Fischer† P. Hofreiter‡ J.-P. Seifert§

Infineon Technologies
Security & ChipCard ICs
CC TI Concepts & Innovations
D-81609 Munich
Germany

Abstract. This article describes concrete results and practically approved countermeasures concerning differential fault attacks on RSA using the CRT. It especially investigates smartcards with a RSA coprocessor where any hardware countermeasure to defeat such fault attacks have been switched off. This scenario has been chosen in order to completely analyze the resulting effects and errors occurring inside the hardware. Using the results of this kind of physical stress attack enables the development of completely reliable software countermeasures. Although *successful* RSA attacks on the investigated hardware have been only possible with an expensive enhanced laboratory equipment, the effects on the unprotected hardware have been tremendously. This caused lots of analysis efforts to investigate what really happened during the attack. Indeed, this will be addressed in this paper.

We first report on the nature of the resulting errors within the hardware due to the physical stress applied to the smartcard. Hereafter, we describe the experiments and results with a very efficient and prominent software RSA-CRT DFA countermeasure. This method could be shown to be insufficient, i.e., detected nearly no error, when we introduced stress at the right position during the computation. Naturally, a detailed error analysis model followed, specifying every failure point during the RSA-CRT operation. This model finally allowed to develop and present here new very practically oriented software countermeasures hedging the observed and characterized fault attacks. Eventually, we present the security analysis of our new developed software RSA-CRT DFA countermeasures. Thanks to their careful specification according to the observed and analyzed errors they resisted all kinds of physical stress attacks and were able to detect any subtle computation error, thus avoiding to break the smartcard by fault attacks.

Nevertheless, we stress, that although our software countermeasures have been fully approved by practical experiments, we are convinced that only sophisticated hardware countermeasures like sensors and filters in combination with software countermeasures will be able to provide a secure and comfortable base to defeat in general any conceivable fault attacks scenario on smartcards properly.

Keywords: Fault attacks, Bellcore attack, Hardware security, Hardware robustness, RSA, Chinese Remainder Theorem, Spike attacks, Transient fault model, Software countermeasures.

* Readers should note that Infineon Technologies Corp. has filed an international patent application containing this work.

** christian.aumueller@infineon.com
*** peter.bier@infineon.com
† wieland.fischer@infineon.com
‡ peter.hofreiter@infineon.com
§ jean-pierre.seifert@infineon.com

1 Introdcution

This paper shows and proves that fault attacks on RSA with the CRT (also known as Bellcore attacks) due to [BDL] are indeed feasible and devastating if there are no hardware mechanisms (like sensors and filters) nor any appropriate software countermeasures implemented in the underlying smartcard ICs. However, this does not imply that modern high-security smartcard ICs are vulnerable by this kind of attacks. Instead, it shows that fault tolerant and robust hardware and especially sophisticated hardware countermeasures are essential for the design of physical secure hardware to prevent such devastating effects as investigated in the following.

Moreover, we stress that it is impossible in the field to switch off these sophisticated hardware countermeasures preventing this kind of attacks — which has been done exceptionally for our detailed feasibility study concerning the practicality of the Bellcore attack by a specifically designed hardware.

In order to provide better security for data protection under strong encryption schemes (e.g., RSA [RSA], 3DES [MvOV], etc.) more and more implementations based on tamper-proof devices (e.g., smartcard ICs) are proposed. The main reason for this trend is that smartcard ICs provide high reliability and security with more memory capacity and better performance characteristics than conventional magnetic stripe card. The CPU in smartcards controls its data input and output and prevents unauthorized access to a smartcard. With special characteristics of computational ability, large memory capacity and security, a large variety of cryptographic applications benefit from smartcard ICs. Due to this popular usage of tamper-resistance, the arising of several new ideas for physical attacks against smartcards in 1996 due to [Koch] and [BDL] and again 1999 by [KJJ], followed by [SQ], has attracted a huge amount of research. However, within this vain, most research so far focused on Timing or Power Analysis attacks. This is surprising as the frauds with smartcards by inducing faults are reality, cf., [A,AK1,AK2], whereas no frauds via Timing or Power Analysis attacks have been reported so far. Moreover, as seen from the scientific literature, the research on faults based cryptanalysis or its practical realization is not very active, when compared with the other side-channel research. Furthermore, no practical investigation of the most interesting and practical scenario, i.e., the so called Bellcore attack [BDL] on RSA with CRT [RSA,CQ] has been reported so far.

Indeed, this topic will be for the first time publicly addressed within this paper. Thus, it answers a question of Kaliski and Robshaw [KR] *of how practical these attacks might be*, answered definitely here *by physicists and the designers and manufactures of secure hardware.*

Actually, the present paper has three main themes. First, we will actually present a practical case study of fault attacks on smartcards implementations of RSA in CRT mode. We will indeed explain how to realize so called spike attacks on smartcards, analyze hereafter their intrinsic complexity from an attacker's point of view and reveal an appropriate test equipment to implement such fault attacks. Second, we will present the resulting errors on completely unprotected hardware and software for RSA in CRT mode. In addition, we will demonstrate the insufficiency of a very prominent and efficient software countermeasure due to [Sh]. Third, we will derive from the analysis of the previously obtained resulting errors new software countermeasures which were proved to fully work under extensive spike attacks.

Only very recently the field of research on fault attacks and their countermeasures has been shown some activity. For instance, a series of papers [YJ,YKLM1,YKLM2,JQYY] came up with some new ideas for attacks scenarios and also new countermeasures to defeat fault attacks. The relevance of this line of research to this paper and especially its practical relevance will be discussed later within section 2.

The present paper is organized as follows. Section 2 briefly repeats RSA using the CRT and the fault based cryptanalysis of RSA using the CRT according to [BDL,JLQ]; it also includes and discusses the advantages and limitations of so far publicly known software countermeasures to defeat fault attacks on RSA in CRT mode. Section 3 basically digs into the real physics of enforcing errors during the cryptographic computation of smartcard ICs. Within section 4 we basically

investigate sophisticated software countermeasures derived from our practical observations and our proposed model to counteract fault attacks on RSA in CRT mode. We also present practical results which were obtained with our countermeasures. Eventually we will draw some practical conclusion in section 5 concerning software countermeasures to hedge such Bellcore attacks.

2 Preliminaries

2.1 The RSA System

Let $N = p \cdot q$ be the product of two large primes each $n/2$ bits long. To sign a message $m \in \mathbb{Z}_N$ using RSA one computes $S := m^d \bmod N$, where d is the private exponent satisfying $e \cdot d \equiv 1 \bmod (p-1)(q-1)$ for the public exponent e. The computationally expensive part of signing using RSA is the modular exponentiation of m. For efficiency most implementations exponentiate as follows: using repeated square and multiply they first compute $S_p := m^d \bmod p$ and hereafter $S_q := m^d \bmod q$. They then use the CRT to construct the signature $S = m^d \bmod N$. This last CRT step takes negligible time compared to with the two exponentiations. It is done efficiently by computing

$$S = S_q + \big((S_p - S_q) * (q^{-1} \bmod p) \bmod p\big) * q, \tag{1}$$

using Garner's algorithm, cf. [Kn].

The reason to use the CRT is that the exponentiation using the CRT is much faster than square and multiply $\bmod N$. To see this, observe that $S_p = m^d \bmod p = m^{d \bmod (p-1)} \bmod p$. Usually, d is of order N, while $d \bmod (p-1)$ is of order p. Consequently, computing S_p requires half as many multiplications as computing S directly. In addition, intermediate values during the computation of S_p are only half as big — they are in the range $[1, \ldots, p]$, rather than $[1, \ldots, N]$. Clearly, the same arguments are valid for the computation of S_q. When quadratic time complexity is used, multiplying two numbers in $\mathbb{Z}_p$ takes a quarter of the time as multiplying elements in $\mathbb{Z}_N$. Hence, computing S_p takes an eight of the time of computing S directly. Thus, computing S_p and S_q this way takes a quarter of the time of computing S directly. Thus, CRT exponentiation is four times faster than direct exponentiation. This is why RSA with CRT is the preferred method for generating RSA signatures, cf. [CQ,MvOV].

2.2 The fault-based cryptanalysis of RSA using CRT

For the sake of completeness we briefly recall the fault-based cryptanalysis of RSA using the CRT due to [BDL] and assume the above notations.

Assume that during the computation of a RSA signature for a message m via the CRT a random error occurs during the computation of S_p, thus yielding the faulty signature part S'_p, whereas the computation of S_q is performed correctly. Applying now the combination of S'_p and S_q via (1) will yield an incorrect signature S' different from the correct signature S, i.e., $S - S' \neq 0$. According to the fault-based cryptanalysis of [BDL,JLQ], one obtains the factorization of N by computing

$$\gcd\left((m - (S')^e) \bmod N, N\right) = q.$$

2.3 Simple software countermeasure to defeat the fault attack

We will now present some simple ad-hoc countermeasures which have been already suggested within [BDL,KR] to hedge the faults attack scenario. One approach is to perform calculations twice and the other approach suggests to verify the correctness of the signature by comparing the inverse result with the input.

The first approach is a very time-consuming and it cannot always provide a satisfactory solution since a permanent error (caused by a permanenet hardware or software fault implementation bug) may be undetectable, even if the function is computed more than once.

The second approach is to verify the correctness by comparing the inverse result with the input m. A RSA signature $S = m^d \bmod N$ can be verified by raising S to the eth power and compare whether $m \equiv s^e \bmod N$. Generally, this is not a satisfactory solution since the parameter e could be a large integer and this checking procedure becomes time-consuming.

A completely different but very interesting countermeasure is the introduction of randomness into the RSA signature process. Here, RSA is applied to $F(m, r)$ where F is some formatting function and r is a random string which ensures that the user never signs the same message twice. Furthermore, given en erroneous signature the verifier does not know the full plain-text that was signed. Consequently, the above attack cannot be applied to this modified system cf. [BDL,BR,KR].

2.4 Shamir's software Countermeasure

Shamir's basic idea, as presented in [Sh] is to select a random integer t and to do the following computations

$$S_{pt} := m^d \bmod p * t,$$
$$S_{qt} := m^d \bmod q * t.$$

In the case of $S_{pt} = S_{qt} \bmod t$ the computation is defined to be error free and S is computed according to the CRT recombination equation (1).

One drawback in Shamir's method, as pointed out in [JPY], is the following. Within the CRT mode of real RSA applications the value d is not known, only the values $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$ are known. Although d can be efficiently computed from d_p and d_q only, as described in [FS], it will definitely limit the acceptance of Shamir's method. Nevertheless, this simple way of checking the computations correctness is anyway insufficient as demonstrated later by our practical experiments.

But, the new software countermeasures, as presented later, developed and motivated by our experimental results alleviate the two formerly drawbacks of Shamir's method.

2.5 General remarks on methods to overcome fault attacks

Only very recently the field of research on fault attack countermeasures has been emerged. For instance a series of papers [YJ,YKLM1,YKLM2,JQYY] assumed that the attacker has a very precise knowledge about the implementation details and especially an absolute accurate control on the timing of his fault induction. This possibility, together with an implemented signature correctness check was then used by the above papers to get access to the bits of the private exponent d. However, all the fault attacks described in [YJ,YKLM1,YKLM1,JQYY] can be easily prevented by the various randomization possibilities for the RSA signature algorithm, e.g., randomization of the message m, modul N and the private exponents d_p and d_q, or simply d. Note that these techniques must be anyway implemented in a secure RSA signature algorithm to counteract other side-channel attacks.

Moreover, [YKLM1] proposed the following very interesting countermeasure. Their key idea is the assurance to influence the computation of S_q or the overall computation of S when an error occurred during the computation of S_p, or vice versa. By the explanation of the cryptanalysis given in section 2 it follows, that in this case no successful fault attack on RSA with CRT is possible. They proposed this idea in order to overcome the problem that an attacker might simply jump over the special critical point where the decision concerning the computation's correctness is done. Although such an attack seems questionable, it can be simply defeated by a small appropriate software. Unfortunately, recently it was shown by [BMS] that their proposal to perform a so called infective RSA CRT computation is not secure, i.e. can be broken completely by lattice reduction methods.

3 Physical fault attacks realization

First of all, we would like to stress again that modern high-end cryptographic devices, e.g., smartcards, are usually protected by means of various and numerous sophisticated hardware mechanisms to detect any intrusion attempt into their system behavior, cf. [Ma,NR]. This is due to the fact that hardware manufacturers of cryptographic devices such as smartcard ICs have been aware of the importance of protecting against intrusions by, e.g., external voltage variations, external clock variations, etc. for a long time. However, it should be clear that the design of such mechanisms is a very difficult engineering task. Such mechanisms must be able to tolerate natural slight deviations from the standard values of the electrical parameter to be safeguarded, in order to ensure proper functionality of the underlying device within the specified range, as for example described in [ISO]. And, on the other side they have to detect very fast and low unnatural deviations from the specified standard range, in order to detect any attack attempt by modifying the electrical execution conditions to alter a computation's result. For example, the standard specification for smartcard ICs [ISO] allows for the smartcard ICs contact V_{CC} under normal operating conditions a voltage supply between $4,5$V and $5,5$V.

Although there are lots of possibilities to introduce an error during the cryptographic operation of an unprotected smartcard hardware, i.e., the CPU working in concert with a crypto coprocessor, we will only explain en detail so called spikes attacks. The reason is that spike attacks are non invasive attacks, which require especially no physical opening and no chemical preparation of the smartcard IC. Thus, spike attacks are the most obvious method for attacking smartcard ICs. For further information on various methods how to enforce erroneous computations of chips without hardware countermeasures, we refer to [A,AK1,AK2,Gu1,Gu2,Koca,Ma].

Spikes As seen above, a smartcard must be able to tolerate on the contact V_{CC} a supply voltage between $4,5$V and $5,5$V, where the standard voltage is specified at 5V. Within this range the smartcard will be able to work properly. However, a deviation of the external power supply of much more than the specified 10% tolerance could cause problems for a proper functionality of the smartcard IC. Indeed, it could then lead to a wrong computation result, provided that the smartcard IC is still able to finish its computation completely. But most often this is not possible, as the spike caused too much trouble to the CPU of the smartcard IC.

Although a spike seems from the above explanation very simple, a specific type of a power spike is determined by altogether nine different parameters. These nine parameters are determined by a combination of time and voltage values and as well by the shape of the transition as shown in figure 1. This indicates the range of different parameters which must be scanned for penetration attacks against cryptographic devices. However, it also reveals the strong requirements for the corresponding sensor mechanisms.

From the former discussion of spike attack, one can envision the difficulties an attacker is confronted with, when he wants to overcome all the activated hardware countermeasures within modern high-security smartcard ICs. Amongst them, there a various, numerous and especially finely tuned sensors and filters monitoring the frequency, voltage supply, etc., designed via highest sophisticated electronically mechanisms.

In the field of PayTV there exist lots of different penetration attacks. For instance, to lock old smartcard devices the TV-channel is used by the TV companies to reprogram the smartcards when connected to the decoder. Thus, after their legal usage time the smartcards are executing an infinity loop. The following figure 2 shows a classical "spike-hardware", which is available from the Internet and is used to crack such locked PayTV smartcards. It does simply spikes on invalidated smartcards in order to leave the programmed infinity loop, which was intended to lock these smartcards. Therefore, these pirate devices are actually called "unlooper"

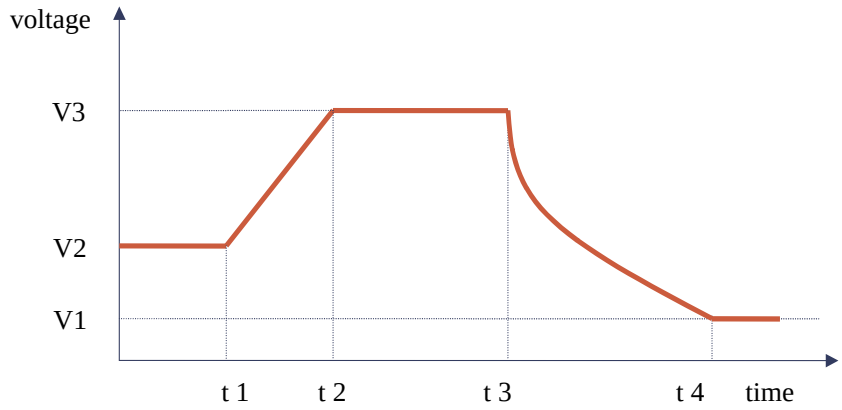

Fig. 1. Spike-parameters defining the shape of a specific spike.

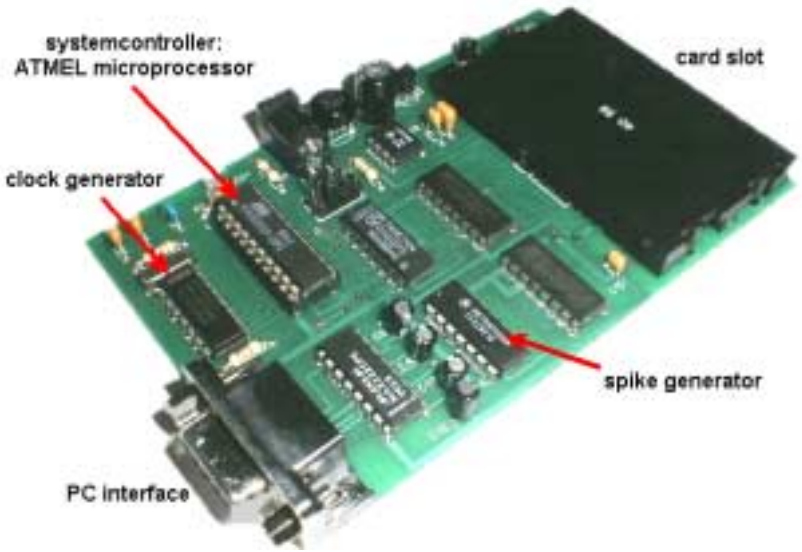

Fig. 2. "Unlooper" hardware from the internet to crack PayTV smartcards.

3.1 Laboratory setting

In order to systematically investigate the effects of spikes and especially our proposed countermeasures, we basically used the following spike enforcing hardware set-up, which is shown in figure 3.

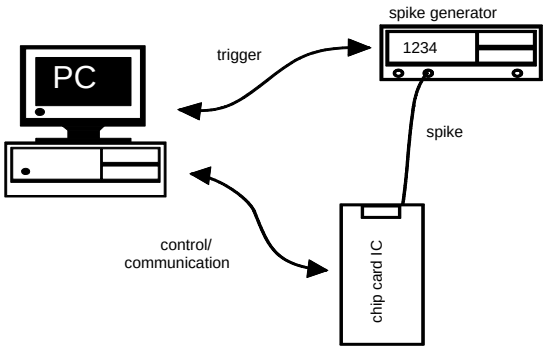

Fig. 3. Schematic of our test equipment.

With such a test set-up it is indeed possible to enforce a spike with a very high preciseness. This is necessary, if the spike shall enforce only a tiny random computation fault rather than a complete destruction of the smartcard's computation, which would make the smartcard's computation result unusable for a successful attack. Through the coupling of the control and communication of the smartcard with a PC, which is running a dedicated test-software, it is possible to observe and analyze the smartcard's reaction with respect to the applied spike-form as discussed above, e.g., answering with a correct/wrong answer sequence. Now, one has to find by altering the 9 parameters of the applied spike a set of parameters enabling a tiny random computation fault, but leaving the smartcard 's main computation untouched.

3.2 Results on unprotected hardware and software

We will now discuss our results of successfully applied spike-attacks on our specifically designed smartcard hardware derived from a typical low-end smartcard. The design modification was basically a disenabling of any hardware countermeasures, to allow fault attacks on RSA using the CRT. Moreover, for ease of exposition, we have also switched off any (hardware and software) countermeasures to defeat other classical side-channel attacks, like Timing Analysis [Koch], Power Analysis [KJJ], Electromagnetic Analysis [SQ], etc.

However, to introduce a spike at the right position of the RSA with the CRT, one first must investigate the power profile of the critical computation. Such a power profile of our investigated coprocessor is shown in figure 4, which we will now explain a little bit more further. The upper line represents the profile of the smartcard's I/O behavior. The first I/O activity is the start impulse for the smartcard and the second peak is the answer sequence given by the smartcard. Between these two peaks the smartcard is computing a 2048-bit RSA signature using the CRT. This is shown in the lower line where the main power profile of the smartcard is depicted.

The RSA-CRT computation starts at the time block 1.5 and ends at the time block 9.2. This can be seen by the fact that the power consumption increases — due to the coprocessors activity. One immediately recognizes the two different exponentiations, as they are the consumers which need for their whole duration the highest power consumption.

In our case the first exponentiation lies in the time frame 1.6 to 5.1, and the second exponentiation lies in the time frame 5.3 to 8.8. Between these two exponentiations there is the loading of

the new data into the crypto coprocessor for the second exponentiation and as well the correctness checks of the first exponentiation. Before the first exponentiation one recognizes the loading of the data into the crypto coprocessor for the first exponentiation, after the first exponentiation the corresponding correctness checks and as well the loading of the data into the crypto coprocessorand for the second exponentiation and after the second exponentiation again the correctness checks of the second exponentiation and finally the CRT combination of the two partial exponentiations followed eventually by additional correctness check for the CRT combination.

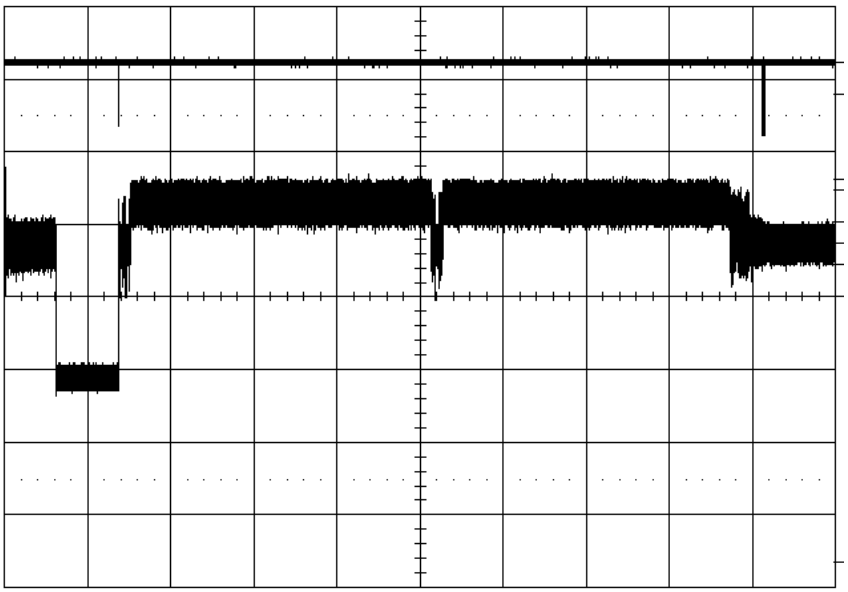

Fig. 4. Power profile of RSA with the CRT.

The first algorithm we attacked with our spike equipment was the pure RSA signature algorithm using the CRT:

input: $m, p, q, d_p, d_q, q^{-1} \bmod p$

$S_p := m^{d_p} \bmod p$
$S_q := m^{d_q} \bmod q$
$S := S_q + ((S_p - S_q) * q^{-1} \bmod p) * q$
return(S)

output: $m^d \bmod N$

Before discussing the results of our spike attacks on the above algorithm we note that the inputs $p, q, d_p, d_q, q^{-1} \bmod p$ are usually stored in EEPROM, while the message m is stored in RAM. However, to compute with the data $p, q, d_p, d_q, q^{-1} \bmod p$ they must moved during the computation from EEPROM into the crypto coprocessor. By varying the time when we applied the appropriate spike to the smartcard ICs power supply V_{CC}, we were able to induce the following different error scenarios.

Observed error scenarios
modification of p, q
modification of d_p, d_q
modification of $q^{-1} \bmod p$
wrong answer sequence of smartcard IC
wrong exponentiation mod$p$
wrong exponentiation mod$q$
faulty signature mod$p$ and mod$q$
error during combination of S_p and S_q

Due to the lack of space we must refer a complete discussion and interpretation of our observed error scenarios to the full version of the present paper. However, from the above table it clearly follows, that an attacker can enforce any error he likes, when hitting the correct time and spike parameters as needed for the underlying unprotected hardware.

Thus, we can conclude that it is absolutely necessary to have sophisticated hardware and software countermeasures to hedge such kinds of attacks to break the RSA signature algorithm using the CRT. Within the remaining sections we will analyze such already existing software countermeasures and also develop new sophisticated and especially more reliable countermeasures.

3.3 Results on unprotected hardware with simple software countermeasures

Motivated by the devastating results obtained within the previous section, we hereafter tested the reliability of the software countermeasures due to [Sh] as desribed in section 2. Thus, we applied at carefully chosen time points our formerly chosen spikes parameters to the unprotected smartcard IC when computing the following RSA signature algorithm, shown in figure 5.

input: $m, p, q, d, q^{-1} \bmod p$

randomly choose a short prime r, e.g., 32 bits
$p' := p * r$
$d'_p := d \bmod (p-1) * (r-1)$
$q' := q * r$
$d'_q := d \bmod (q-1) * (r-1)$

$S'_p := (m \bmod p')^{d'_p} \bmod p'$
$S'_q := (m \bmod q')^{d'_q} \bmod q'$

$S_p := S'_p \bmod p$
$S_q := S'_q \bmod q$
$S := S_q + ((S_p - S_q) * q^{-1} \bmod p) * q$

if $((S'_p \bmod r) = (S'_q \bmod r))$ **then**
  return(S)
else
  return(error)

output: $m^d \bmod (p * q)$

Fig. 5. Shamir's countermeasure.

We will now briefly summarize in a table the observed errors. But again, due to the lack of space we must refer a detailed explanation and discussion of the observed errors, their nature

and especially their security consequences to the full version of the present paper. But to give an impression of possible problems, consider the case that during the computation of p' the value of p is changed to some value $\tilde{p}$, such that $p' = \tilde{p}r$. Then the correctness check mod r will fail.

Observed error scenarios	recog.?	relev.?	working?
modification of p, q	time dep.	time dep.	time dep.
modification of d, d'_p, d'_q	time dep.	no	yes
modification of $q^{-1} \bmod p$	no	yes	no
modification of r	time dep.	time dep.	yes
wrong exp. mod$p$	prob. $1 - 1/t$	yes	yes
wrong exp. mod$q$	prob. $1 - 1/t$	yes	yes
faulty signature mod$p$ and mod$q$	prob. $1 - 1/t$	no	no
error during comb. of S_p and S_q	no	yes	no
modification of S_p or S_q	time dep.	yes	no

The above table is organized as follows. The first column denotes the kind of error which might occur. The second column indicates whether the countermeasure recognizes the induced fault, while the second indicates whether the corresponding type of fault reveals the secret key. Finally, the last column says whether the countermeasure recognizes the devastating faults.

4 Practical Fault attacks countermeasures for unprotected hardware

Within this section we will first analyze the observed error scenarios from the two former sections and hereafter propose our new countermeasures. However, due to the lack of space this section will be very shortened and we again refer the reader to the full version of our paper.

4.1 Model to understand resulting/possible faults

From the observed error scenario, we have learned by an extensive data analysis the following facts.

- During the computation, every input value to the RSA signature algorithm could be altered to a value different from the original value.
- During the computation, every variable can be changed.
- The only values to trust, are the values which are stored in ROM or EEPROM.

Armed with this knowledge, we formulated the following checking philosophy:

Check (at least in a probabilistic sense) every computed intermediate result wrt. its correctness by relying on trusted values only.

In a rough sense, this checking philosophy is reflected by figure 6, showing the old and the new checking philosophy.

4.2 Software countermeasures derived according to the model

Inspired by the previous section and strong efficiency requirements, we developed the following countermeasures to hedge the fault attack scenario on RSA using the CRT. Also it takes into account, that a practical application is given d_p and d_q only, instead of having access to the full d. Also, it avoids the use of the public exponent e, which is most often not known to the signature algorithm in real applications.

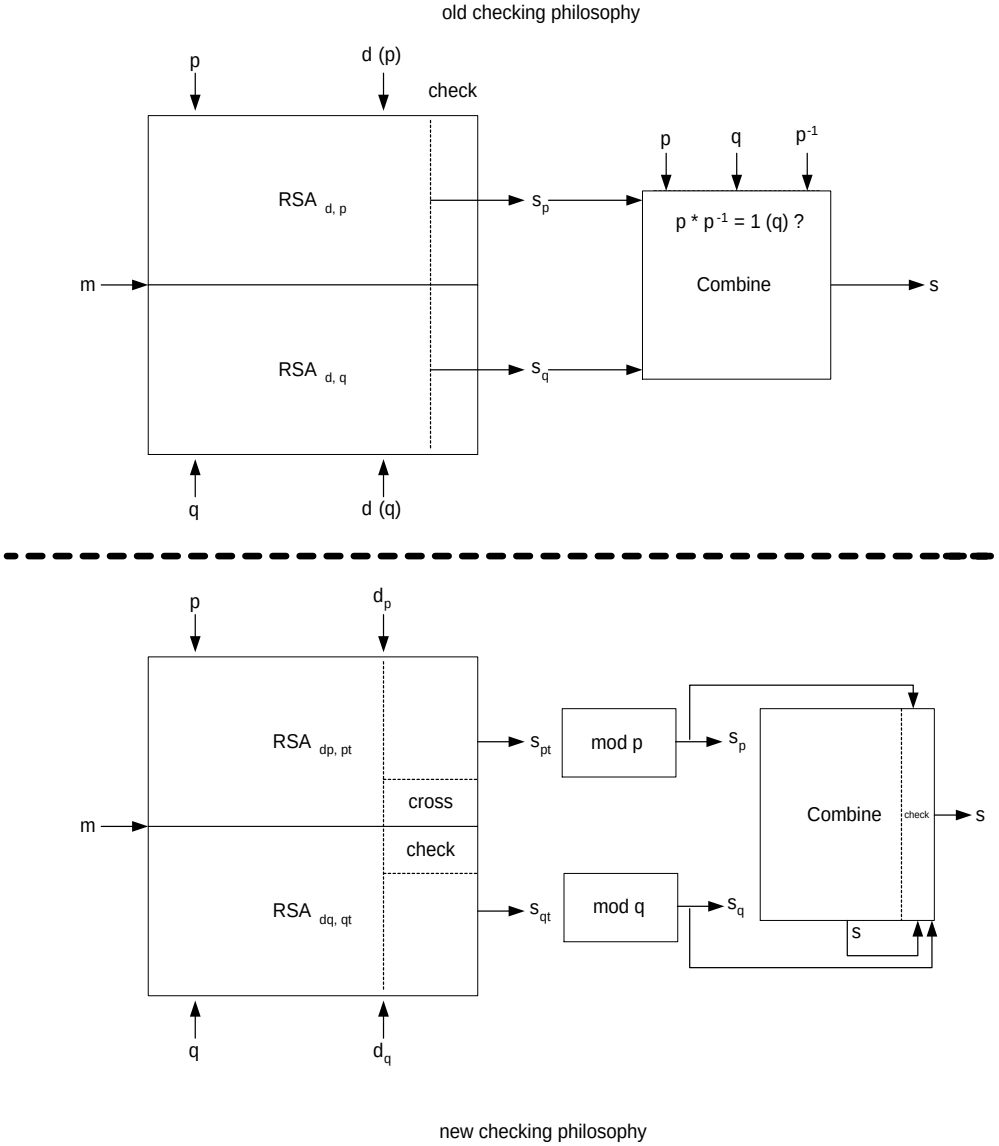

Fig. 6. Information flow during checking.

input: $m, p, q, d_p, d_q, q^{-1} \bmod p$

let t be a short prime number, e.g., 16 bits

$p' := p * t$
$d'_p := d_p + \text{random}_1 * (p-1)$
$S'_p := m^{d'_p} \bmod p'$
if $\neg(p' \bmod p \equiv 0 \ \wedge \ d'_p \bmod (p-1) \equiv d_p)$ **then return**(error)

$q' := q * t$
$d'_q := d_q + \text{random}_2 * (q-1)$
$S'_q := m^{d'_q} \bmod q'$
if $\neg(q' \bmod q \equiv 0 \ \wedge \ d'_q \bmod (q-1) \equiv d_q)$ **then return**(error)

$S_p := S'_p \bmod p$
$S_q := S'_q \bmod q$
$S := S_q + ((S_p - S_q) * q^{-1} \bmod p) * q$
if $\neg((S \bmod p = S_p) \ \wedge \ (S \bmod q = S_q))$ **then return**(error)

$S_{pt} := S'_p \bmod t$
$d_{pt} := d'_p \bmod (t-1)$
$S_{qt} := S'_q \bmod t$
$d_{qt} := d'_q \bmod (t-1)$
if $(S_{pt}^{d_{qt}} \equiv S_{qt}^{d_{pt}} \bmod t)$ **then**
  return(S)
else
  return(error)

output: $m^d \bmod (p * q)$

Fig. 7. Practically secured RSA with CRT.

4.3 Measurement results for enhanced software countermeasures

Through extensive penetration tests via spikes on the algorithm shown in figure 6 we obtained the following table proving empirically the reliability of our software countermeasures.

Observed error scenarios	recog.?	relev.?	working?
modification of p, p', q, q'	yes	yes	yes
modification of d'_p, d'_q	yes	yes	yes
modification of $q^{-1} \bmod p$	yes	yes	yes
modification of t	yes	yes	yes
wrong exp. mod$p$	prob. $1 - 1/t$	yes	yes
wrong exp. mod$q$	prob. $1 - 1/t$	yes	yes
faulty signature mod$p$ and mod$q$	prob. $1 - 1/t$	no	yes
error during comb. of S_p and S_q	yes	yes	yes
modification of S_p or S_q	yes	yes	yes

Clearly, the probability that an error is undetected is equal to $1/t$. For t a 64-bit integer, this probability is small enough; t can thus be seen as a security parameter.

5 Conclusion

We have shown that the classical RSA with CRT fault attack is in principal feasible when using completely unprotected microcontrollers and moreover, that also prominent and efficient software countermeasures are not always completely reliable. Thus, it answers again a question of Kaliski and Robshaw [KR] to the affirmative, that these attacks are indeed practical. Moreover, our investigation also reveals that one should test any conceivable countermeasures in reality against all possible attack scenarios before trusting them. This was especially done with our newly developed software countermeasures which are indeed practical, efficient and fully approved by extensive practical penetration test. We would like to stress again, that our successful attacks have been only possible by switching off the whole zoo of implemented hardware countermeasures. In the field these mechanisms are always switched on to counteract spike attacks and lots of other attacks in order to give the smartcard user a full functional tamper-resistant device. And indeed, such mechanisms must be implemented on the card to prevent other known attacks rather than counteracting the simple (but efficient) attack on the RSA signature algorithm.

So, we close with an advice due to Kaliski and Robshaw [KR] from the RSA Laboratories that *good engineering practices in the design of secure hardware are essential.*

6 Acknowledgments

We would like to thank Jörg Schepers for helpful discussions.

References

[A] R. Anderson, *Security Engineering*, John Wiley & Sons, New York, 2001.

[AK1] R. Anderson, M. Kuhn, "Tamper Resistance – a cautionary note", *Proc. of 2nd USENIX Workshop on Electronic Commerce*, pp. 1-11, 1996.

[AK2] R. Anderson, M. Kuhn, "Low cost attacks attacks on tamper resistant devices", *Proc. of 1997 Security Protocols Workshop*, Springer LNCS vol. 1361, pp. 125-136, 1997.

[BDL] D. Boneh, R. A. DeMillo, R. Lipton, "On the Importance of Eliminating Errors in Cryptographic Computations" *Journal of Cryptology* **14**(2):101-120, 2001.

[BDHJ+] F. Bao, R. H. Deng, Y. Han, A. Jeng, A. D. Narasimbalu, T. Ngair, "Breaking public key cryptosystems on tamper resistant dives in the presence of transient faults", *Proc. of 1997 Security Protocols Workshop*, Springer LNCS vol. 1361, pp. 115-124, 1997.

[BR] M. Bellare, P. Rogaway, "The exact security of digital signatures — how to sign with RSA and Rabin", *Proc. of EUROCRYPTO '96*, Springer LNCS vol. 1070, pp. 399-416, 1996.

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

[BMM] I. Biehl, B. Meyer, V. Müller, "Differential fault attacks on elliptic curve cryptosystems", *Proc. of CRYPTO '00*, Springer LNCS vol. 1880, pp. 131-146, 2000.

[BMS] J. Blömer, A. May, J.-P. Seifert, personal communication, April 2002.

[CQ] C. Couvreur, J.-J. Quisquater, "Fast decipherment algorithm for RSA public-key cryptosystem", *Electronics Letters* **18**(21):905-907, 1982.

[FS] W. Fischer, J.-P. Seifert, "Note on fast computation of secret RSA exponents", *Proc. of ACISP '02*, Springer LNCS vol. ?, pp. ?-?, 2002.

[Gu1] P. Gutmann, "Secure deletion of data from magnetic and solid-state memory", *Proc. of 6th USENIX Security Symposium*, pp. 77-89, 1997.

[Gu2] P. Gutmann, "Data Remanence in Semiconductor Devices", *Proc. of 7th USENIX Security Symposium*, pp. ?-?, 1998.

[HP1] H. Handschuh, P. Pailler, "Smart Card Crypto-Coprocessors for Public-Key Cryptography", *CryptoBytes* **4**(1):6-11, 1998.

[HP2] H. Handschuh, P. Pailler, "Smart Card Crypto-Coprocessors for Public-Key Cryptography", *Proc. of CARDIS '98*, Springer LNCS vol. 1820, pp. 372-379, 1998.

[ISO] International Organization for Standardization, "ISO/IEC 7816-3: Electronic signals and transmission protocols", `http://www.iso.ch`, 2002.

[JLQ] M. Joye, A. K. Lenstra, J.-J. Quisquater, "Chinese remaindering based cryptosystem in the presence of faults", *Journal of Cryptology* **12**(4):241-245, 1999.

[JPY] M. Joye, P. Pailler, S.-M. Yen, "Secure Evaluation of Modular Functions", *Proc. of 2001 International Workshop on Cryptology and Network Security*, pp. 227-229, 2001.

[JQBD] M. Joye, J.-J. Quisquater, F. Bao, R. H. Deng, "RSA-type signatures in the presence of transient faults", *Cryptography and Coding*, Springer LNCS vol. 1335, pp. 155-160, 1997.

[JQYY] M. Joye, J.-J. Quisquater, S. M. Yen, M. Yung, "Observability analysis — detecting when improved cryptosystems fail", *Proc. of CT-RSA Conference 2002*, Springer LNCS vol. 2271, pp. 17-29, 2002.

[KR] B. Kaliski, M. J. B. Robshaw, "Comments on some new attacks on cryptographic devices", *RSA Laboratories Bulletin* **5**, July 1997.

[Kn] D. E. Knuth, *The Art of Computer Programming, Vol.2: Seminumerical Algorithms*, 3rd ed., Addison-Wesley, Reading MA, 1999.

[Koca] O. Kocar, "Hardwaresicherheit von Mikrochips in Chipkarten", *Datenschutz und Datensicherheit* **20**(7):421-424, 1996.

[Koch] P. Kocher, "Timing attacks on implementations of Diffie-Hellmann, RSA, DSS and other systems", *Proc. of CYRPTO '97*, Springer LNCS vol. 1109, pp. 104-113, 1997.

[KJJ] P. Kocher, J. Jaffe, J. Jun, "Differential Power Analysis", *Proc. of CYRPTO '99*, Springer LNCS vol. 1666, pp. 388-397, 1999.

[Ma] D. P. Maher, "Fault induction attacks, tamper resistance, and hostile reverse engineering in perspective", *Proc. of Financial Cryptography*, Springer LNCS vol. 1318, pp. 109-121, 1997.

[MvOV] A. J. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, New York, 1997.

[NR] D. Naccache, D. M'Raihi, "Cryptographic smart cards", *IEEE Micro*, pp. 14-24, 1996.

[Pe] I. Petersen, "Chinks in digital armor — Exploiting faults to break smartcard cryptosystems", *Science News* **151**(5):78-79, 1997.

[RSA] R. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Comm. of the ACM* **21**:120-126, 1978.

[SQ] D. Samyde, J.-J. Quisquater, "ElectroMagnetic Analysis (EMA): Measures and Countermeasures for Smart Cards", *Proc. of Int. Conf. on Research in Smart Cards, E-Smart 2001*, Springer LNCS vol. 2140, pp. 200-210, 2001.

[Sh] A. Shamir, "Method and Apparatus for protecting public key schemes from timing and fault attacks", U.S. Patent Number 5,991,415, November 1999; also presented at the rump session of EUROCRYPT'97.

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

[YKLM1] S.-M. Yen, S.-J. Kim, S.-G. Lim, S.-J. Moon, "RSA Speedup with Residue Number System immune from Hardware fault cryptanalysis", *Proc. of the ICISC 2001*, Springer LNCS vol. ?, pp. ?-?, 2001.

[YKLM2] S.-M. Yen, S.-J. Kim, S.-G. Lim, S.-J. Moon, "A countermeasure against one physical cryptanalysis may benefit another attack", *Proc. of the ICISC 2001*, Springer LNCS vol. ?, pp. ?-?, 2001.

[ZM] Y. Zheng, T. Matsumoto, "Breaking real-world implementations of cryptosystems by manipulating their random number generation", *Proc. of the 1997 Symposium on Cryptography and Information Security*, Springer LNCS vol. ?, pp. ?-?, 1997.