A DPA Attack against the Modular Reduction within a CRT Implementation of RSA

Bert den Boer*, Kerstin Lemke, and Guntram Wicke

T-Systems ISS GmbH
Rabinstr. 8, D-53111 Bonn, Germany
BdenBoer@tpd.tno.nl, {Kerstin.Lemke,Guntram.Wicke}@t-systems.com

Abstract. Published DPA attack scenarios against the RSA implementation exploit the possibility of predicting intermediate data during a straight-forward square-multiply exponentiation algorithm. An implementation of RSA using CRT (Chinese Remainder Theorem) prevents the pre-calculation of intermediate results during the exponentiation algorithm by an attacker. In this paper, we present a DPA attack that uses byte-wise hypotheses on the remainder after the modular reduction with one of the primes. Instead of using random input data this attack uses k series of input data with an equidistant step distance of 1, 256, $(256)^2$, ..., $(256)^k$. The basic assumption of this DPA attack named MRED ("Modular Reduction on Equidistant Data") is that the distance of the input data equals the distance of the intermediate data after the modular reduction at least for a subgroup of single measurements. A function F_k that is composed of the k DPA results is used for the approximation of a multiple of the prime. Finally the gcd gives the prime. The number of DPA calculations increases linear to the number of bytes of the prime to be attacked. MRED is demonstrated using simulated measurement data. The practical efficiency is assessed. If the applicability of this attack is limited due to padding formats in RSA signature applications, the least significant bytes of the remainder after the modular reduction step can still be revealed. Multiplicative message blinding can protect the reduction modulo a secret prime against MRED.

Keywords. DPA, modular reduction, CRT, RSA, power analysis, side channel analysis, smartcard

1 Introduction

For the last years an increased research is focused on vulnerabilities of implementations of cryptographic algorithms. These vulnerabilities in the 'real world' applications are e. g. caused by the deterministic dependencies of the power consumption, electromagnetic radiation and timing characteristics on the processed data. Generally, these attacks don't leave any visible damage to the cryptographic module that can be recognised by the users. Besides that, there are

* now at: TNO TPD, PO Box 155, NL-2600 AD Delft, The Netherlands

B.S. Kaliski Jr. et al. (Eds.): CHES 2002, LNCS 2523, pp. 228–243, 2003.

fault analysis attack scenarios that aim to cause transient or permanent faults during the cryptographic calculation that can be exploited mathematically ([9], [10]). These kinds of attacks yield a new field of attacks on cryptographic algorithms. They are generally summarised as 'Side Channel Cryptanalysis' in contrast to the mathematical cryptographic analysis of the algorithm itself.

Processor smartcards that are applied in security relevant applications (e. g. digital signature) are of a special interest. These products have to guarantee that attack scenarios of 'Side Channel Cryptanalysis' that fall into a category of up to high attack potential are prevented effectively.

Power analysis attacks SPA ("Simple Power Analysis") and DPA ("Differential Power Analysis") were first published by P. Kocher et al. [1] and it turned out that the statistical attack DPA is very effective and can be applied without the knowledge of implementation details. DPA attacks were first used to compromise DES keys during the use of the DES algorithm.

The first power analysis attacks on the RSA algorithm were published by Thomas S. Messerges et al. [2]. Attack scenarios SEMD ("Single Exponent, Multiple Data"), MESD ("Multiple Exponent, Single Data") and ZEMD ("Zero Exponent, Multiple Data") were introduced. The ZEMD attack uses DPA techniques to compromise the bits of the private RSA exponent sucessively. This ZEMD attack is applied on the intermediate results during modular exponentiation. A basic precondition of this attack is that the intermediate data of the modular exponentiation can be predicted offline.

DPA attacks against RSA are classified as 'chosen ciphertext' attacks if applied at the RSA decryption. If the DPA attacks are applied against the RSA signature the attacks belong to the 'chosen plaintext' category.

Due to the effectiveness of power and timing analysis on RSA implementations algorithmic countermeasures are introduced to counteract the predictability of intermediate data. So far, a DPA attack on the CRT implementation was not published. Thus, implementations using CRT may rely on the unpredictability of intermediate data because of the modular reduction step that is carried out with one of the secret primes before the modular exponentiation starts.

2 RSA and the CRT Implementation

In this subsection we recollect the well known CRT algorithm.

The RSA cryptosystem is given by the secret RSA primes p and q, the public modulus N with $N = p\,q$, the public exponent e and the secret exponent d with $e\,d \equiv 1(\mathrm{mod}\ \mathrm{lcm}(\mathrm{p}-1, \mathrm{q}-1))$ as its parameters and the operations for decryption $y = x^d \bmod N$ and encryption $x = y^e \bmod N$.

Widely used techniques to perform the decryption operation are 'square and multiply' algorithms in conjunction with techniques using the Chinese Remainder Theorem (CRT) for known secret primes p and q.

To perform a modular exponentiation $c = a^b \bmod m$ in $\mathbb{Z}_m$, the bitwise representation $b = [b_{n-1}b_{n-2}\cdots b_1 b_0]$ is used. The 'square - multiply' algorithm

evaluates this representation either starting from the least significant bit b_0 (algorithm A1) or from the most significant bit b_{n-1} (algorithm A2).

A1:

```
t := a
c := 1
for k := 0 to n-1 do {
  if b[k]=1 then c := c*t mod m
  t := t*t mod m
}
return c
```

A2:

```
c := 1
for k := n-1 down to 0 do {
  c := c*c mod m
  if b[k]=1 then c := c*a mod m
}
return c
```

To reduce calculation time of a RSA exponentiation with the secret key one can solve a simultaneous system of modular congruencies. The existence of such a solution is ensured by the Chinese Remainder Theorem (CRT). We follow the common practice and denote also an algorithm that solves modular congruencies using the theorem as a CRT algorithm. The special case of RSA requires the representation of the exponentiation in terms of the RSA primes p and q and their recombination to the solution mod$N$. The calculation is then about four times faster than an exponentiation mod$N$.

Fermat's little theorem allows the precalculation of the reduced secret exponent values $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$. Using A1 or A2 one calculates then $v_1 = x^{d_p} \bmod p$ and $v_2 = x^{d_q} \bmod q$. For the CRT algorithm according to Garner (A3) one precalculated multiplicative inverse $P_q = p^{-1} \bmod q$ is needed:

A3:

```
u := (v2-v1)*Pq mod q
y := v1+u*p
return y
```

Alternatively, the CRT algorithm according to Gauss (A4) uses the two precalculated multiplicative inverses $P_q = p^{-1} \bmod q$ and $Q_p = q^{-1} \bmod p$ and a final reduction modulo N:

A4:

```
y := (v1*q*Qp + v2*p*Pq) mod N
return y
```

Because of memory constraints implementations on smartcards generally prefer the usage of algorithm A2 and A3. Note that during exponentiation a modular reduction modulo a secret value instead of a public one takes place. This is used for the attack described below.

3 DPA against a Non-CRT Implementation

To apply DPA to RSA the attacker should have the possibility to randomly vary the input data x of the RSA implementation to be attacked. Single power consumption measurements $P(x, t)$ of the cryptographic module are typically carried out with a digital oscilloscope and stored on a file server or PC.

3.1 Key Hypotheses

If the RSA implementation uses a straigthtforward 'top-down square-multiply' algorithm (A2 of section 2) the key hypotheses are set up on the next bits of the exponent to proceed. The intermediate results of the exponentiation algorithm proceeding these bits can be determined offline. E. g. in the simplest case the attacker uses only two hypotheses, namely

1. 'the next exponent bit is 0' and
2. 'the next exponent bit is 1'.

In case that the second hypothesis is correct, correlations are present for both key hypotheses as the result of the first hypothesis is an intermediate result of the second hypothesis. The correlations for the correct key hypothesis appear last.

The number of exponent bits used can be optimised under the limitations that taking a bigger number of bits increases the computation time. Generally, it is even more useful to set up the key hypotheses on the sequence of elementary operation (squarings 'Q' and multiplications 'M'). The simplest hypotheses would be

1. 'the next 2 modular multiplication units are composed of 'QM',
2. 'the next 2 modular multiplication units are composed of 'QQ', and
   – in case that the previous correct hypothesis ends up with a 'Q' – additionally
3. 'the next 2 modular multiplication units are composed of 'MQ'.

In general, this set-up of key hypotheses is of interest if we deal with a greater number of key hypotheses. The time window in which correlations are expected can be limited to 1-2 elementary modular multiplication units.

3.2 Selection Functions

Power Analysis is based on the dependency of the power consumption, used by the hardware, on the value of intermediate data. The attacker knows or assumes a model for this dependency. A common model is that the power consumption correlates with the Hamming weight of intermediate data (see [3], [4], [5]).

The selection function has to be calculated on the intermediate result of each key hypothesis that is applied. Intermediate results of the RSA exponentiation are generally of the same bit length as the modulus used. The n-bit bus architecture of the RSA coprocessor used determines the number of bits that are taken into account for the Hamming weight. Common bus widths of cryptographic RSA coprocessors are 32 and 64 bit. It is not necessary that an attacker knows the precise internal bus width. It can be tested using DPA. DPA selection functions $d(x)$ should use the bit-width of the bus architecture to set up functions on the Hamming weight $W(x)$ of intermediate data. A simple selection function $d(x)$ assesses all intermediate data values that have a greater Hamming weight than the n-bit expectation value $E(n) = n/2$ with 1, all values with smaller Hamming weights than $E(n)$ with -1 and to ignore all values that meet the expectation value $E(n)$.

$$d(x) = \begin{cases} -1, \text{ if } W(x) < E(n) \\ 0, \text{ if } W(x) = E(n) \\ +1, \text{ if } W(x) > E(n) \end{cases} \tag{1}$$

Translating towards the values $\{-1, 0, 1\}$ looses information. The selection function $d(x)$ can be refined in the linear model to be

$$d(x) = W(x) - E(n). \tag{2}$$

The easiest selection function is the Hamming weight of intermediate data, or for example the Hamming weight of just a byte of transported data.

3.3 Correlation

DPA identifies the correct key hypothesis by assessing the absolute maximum of the correlation coefficients for each key hypothesis. The correlation is carried out between the result of the selection function $d(x, j)$ on the base of the key hypothesis j and the input data x and the power consumption $P(x, t)$ of the single measurements as a function of x and the time t. The variable t could be narrowed to a small time interval if simple power characteristics of the implementation are obvious. The number i runs through all single measurements.

$$c(t, j) = \frac{\sum_i (d(x_i, j) - \overline{d(x_i, j)})(P(x_i, t) - \overline{P(x_i, t)})}{\sqrt{\sum_i (d(x_i, j) - \overline{d(x_i, j)})^2} \sqrt{\sum_i (P(x_i, t) - \overline{P(x_i, t)})^2}} \tag{3}$$

The correlation coefficient $c(t, j)$ has to be assessed for each key hypothesis j. It will be near zero if there aren't any correlations between the selection function

$d(x, j)$ and $P(x, t)$. In case of a strong correlation $c(t, j)$ approaches 1 at some specific points in time. If there are significant correlation results at a certain key hypothesis j that do not occur at other key hypotheses this is a strong indication of the correct key values.

The formula (3) gives an insight in the notion of (cross)-correlation, but for efficient computation the formula should be reordered.

4 DPA Attack against a CRT Implementation

For performance reasons the CRT implementation is a common choice of a RSA implementation. If the Chinese Remainder Theorem is used intermediate data of the modular exponentiation algorithm are unknown, as the input value of the modular exponentiation algorithm is reduced modulo the primes p and q, respectively. The offline-prediction of intermediate data during the square-multiply algorithm is not possible anymore.

4.1 Basic Idea: Hypotheses on the Remainder

In contrast to the ZEMD that has to correlate on the intermediate results of the modular exponentiation algorithm this DPA attack on the CRT implementation attacks the modular reduction modulo one of the primes performed prior to the CRT exponentiation. It exploits power consumption signals that are caused by the processing and data bus transfers of the residue.

The DPA attack on the CRT implementation uses measurement series with input values of RSA that are equidistant. It assumes that input values can be chosen by the attacker. At the first measurement series a starting value x_0 is chosen and the following input values are generated by decrementing the previous input value by 1. We assume that each series contains m elements. Series are numbered with k. Within the second series the input values have a distance of 256: x_0, $x_0 - 1 \cdot 256$, $x_0 - 2 \cdot 256$, $x_0 - 3 \cdot 256$, ..., $x_0 - m \cdot 256$. Other series follow with stepsize 256^k until the exponent k reaches the size of the prime to be attacked.

There aren't any further restrictions on the input value x_0. We can deal with a purely random, modulus-sized number. For each series k we define the i-th value

$$x_i = x_0 - i \cdot (256)^k \,. \tag{4}$$

The DPA attack on the modular reduction sets up hypotheses on the remainder r after the reduction modulo the prime q. The aim of the first measurement series with the distance 1 of the input values is to compromise the least significant byte of the remainder r_0 that fulfills

$$r_0 \equiv x_0 \bmod q \,. \tag{5}$$

We further define

$$r_i = x_i \bmod q\,. \tag{6}$$

The further measurement series aim to compromise the k-th byte of the remainder r_0, respectively.
For demonstration purposes we focus first on the usage of the measurement series with an equal distance of 1 before we give the general approach.

We assure that all input values x_i of the first measurement series have a remainder with the prime (in the following we assume that the prime q is going to be attacked) that does not equal zero. We therefore exclude the unlikely case of crossing a multiple of q by calculating the greatest common divisor with the public modulus N and all input values x_i of the first measurement series: $gcd(x_i, N)$. In the unlikely case that a multiple of q is crossed the modulus N is directly factorised.

There are 256 hypotheses $H_{j0} (0 \leq j \leq 255)$ on the value of the least significant byte of r_0

$$H_{j0} \;\text{ is }\; \{r_0 \bmod 256 = j\} \tag{7}$$

that are going to be analysed with DPA.

All values of r_i are related to the value r_0. As the input values x_i are equally distant the difference between r_0 and r_i directly gives the value of the last byte of the remainder for each hypothesis

$$H_{ji} \;\text{ is }\; \{r_i \bmod 256 = (j - i) \bmod 256\}\,. \tag{8}$$

The corresponding value of H_{ji} can be read out from the Table 1.

Table 1. Table of hypotheses H_{ji}

H_{ji}	x_0	x_1	x_2	x_3	x_4	$\cdots$	x_i
H_{0i}	0	255	254	253	252	$\cdots$	$-i \bmod 256$
H_{1i}	1	0	255	254	253	$\cdots$	$(1 - i) \bmod 256$
H_{2i}	2	1	0	255	254	$\cdots$	$(2 - i) \bmod 256$
.....	...	...	...	...	...	..	.....
H_{255i}	255	254	253	252	251	$\cdots$	$(255 - i) \bmod 256$

The correlation is carried out with the Hamming weight $W(x)$ for each hypothesis H_{ji}. The selection function $d(x, j)$ is therefore based on 8 bit (see Table 2).

The strongest results are expected for that value of j where the hypothesis corresponds to the reality. This value is called f_0 from now on. The cyclic property of H_{ji} yields secondary correlation peaks. The second strongest correlations are expected at the hypothesis $H_{j\pm 128}$. The third strongest correlations should be at the two hypotheses $H_{j\pm 64}$. Therefore there are additional indices of the correct hypothesis.

Table 2. Table of the selection functions $d(x_i, j)$ on the base of hypotheses H_{ji} using the 8-bit Hamming weight $W(x)$.

d_{ji}	x_0	x_1	x_2	x_3	x_4	$\cdots$	x_i
d_{0i}	0	8	7	7	6	$\cdots$	$W(H_{0i})$
d_{1i}	1	0	8	7	7	$\cdots$	$W(H_{1i})$
d_{2i}	1	1	0	8	7	$\cdots$	$W(H_{2i})$
....	..	..	..	..	..	..	.....
d_{255i}	8	7	7	6	7	$\cdots$	$W(H_{255i})$

As result of the first measurement series it is found that $(x_0 - f_0) \bmod q$ is divisible by 256.

4.2 The General DPA Attack: MRED

Accordingly to the first measurement series the attack scenarios on the more significant bytes of the remainder r_0 are carried out. k denotes the current byte that is attacked with DPA and the least significant byte of the remainder is referenced with $k = 0$.

The reference base of each measurement series for the successive approximation of r_0 remains x_0. All other bases are calculated by a decrement of $(256)^k$. The 256 hypotheses used for DPA are

$$H_{ji} \text{ is } \left\{ (r_i \bmod (256)^{k+1}) \operatorname{div} (256^k) = (j - i) \bmod 256 \right\}. \tag{9}$$

We define $F_k = r_0 \bmod (256)^k$. With the discovery f_{k-1} of the previous measurement series the function F_k is given by

$$F_k = \sum_{i=0}^{k-1} f_i \cdot (256)^i . \tag{10}$$

The pre-condition for the cyclic DPA attack is that

$$(x_0 - i \cdot (256)^k) \bmod q = r_0 - i \cdot (256)^k . \tag{11}$$

Whether it holds for all m elements of the measurement series depends on the fact whether $r_0 \geq m \cdot (256)^k$. Because of formula (10) this is equivalent to $(r_0 - F_k) \geq m \cdot (256)^k$. If this is not true, then there exists a $w \leq m$ such that $(r_0 - F_k) = w \cdot (256)^k$. This last equality implies that $(x_0 - F_k) = (w \cdot (256)^k) \bmod q$. To test whether such a $w \leq m$ has occurred, it is checked whether $((x_0 - F_k) - i \cdot (256)^k)$ was divisible by q for one of the candidate values $i \leq m$. This testing is done by computing the gcd of all elements of a measurement series with the modulus N and the check whether it is 1:

$$gcd(x_0 - F_k - i \cdot (256)^k, N) \stackrel{!}{=} 1. \tag{12}$$

Otherwise, the modulus N is factorised as a result of the gcd and this is the end criterion of this DPA attack. The check can be optimised by computing the product of the elements modulo N, comparing with 0 at each step and one final gcd calculation.

Then we examine the measurements on $x_0 - i \cdot (256)^k$ with DPA methods to find f_k, which means that $(x_0 \bmod q) \bmod (256)^{(k+1)}$ equals $F_{k+1} = f_0 + f_1 \cdot 256 + \cdots + f_k \cdot (256)^k$.

This procedure of alternating gcd calculation and DPA calculation finds more and more bytes from r_0, starting from the least significant byte. The function $x_0 - F_k$ continously approaches a multiple of the prime q starting from the least significant byte. Along this line the measurements can be done beforehand, while the alternate DPA search and gcd calculation in the end finds q.

The basic assumption for a successful DPA attack is formula (11). This DPA attack that is referred to "MRED" (Modular Reduction using Equidistant Data) is applicable if this equality holds at least for a certain percentage of single measurements.

MRED needs up to q_n measurement series whereas q_n is the byte length of the prime q. Each measurement series typically has to include a few hundred to a few thousand single measurements. The measurements can be taken in advance before the DPA calculations.

5 Results

The results that are expected using this DPA attack on the CRT implementation are demonstrated using simulated measurement data. The generation of these data is based on the power leakage model that the power consumption $P(x,t)$ at a certain point in time t can be split into a power contribution that varies with the Hamming weight of the data x processed, into a power consumption that represents a constant portion and a power consumption that is caused by noise [6][7]. The simulation data are generated using $P(x,t) = P'(x,t) + N(t)$, whereas $P'(x,t)$ is deterministic and depends on the Hamming weight of the data. $N(t)$ simulates a random noise level and should have zero mean. In the linear model $P'(x,t)$ is proportional to the Hamming weight $W(x)$ of the intermediate data according to the expectation value $E(n)$:

$$P(x,t) = (W(x) - E(n)) \cdot \Delta(t) + R(t) + N(t). \tag{13}$$

$\Delta(t)$ is a certain portion of power consumption for each bit transported that does not equal zero in data dependent paths. R(t) is the remaining deterministic part. Noise $N(t)$ can be ignored at statistical attacks [6].

The underlying bus-architecture is chosen to be 8 and 32 bit, respectively. The generation of simulated measurement data gives an output file for each exponentiation that contains the Hamming weight of all intermediate data processed. These output files replace the single measurement data files. The number of bits used for the calculation of the Hamming weight is given by the bus-architecture.

The starting value x_0 was chosen randomly as 128 byte value. The value of prime q was 63 byte long.

The test values used are the following.

q:

```
00 DA 2B AD CF F0 83 45 0E 4D 8F 32 EF 68 3A 57
06 DB E5 2E 15 8B 8F 9F 62 4C 15 D8 91 B9 03 56
B5 FB B8 35 88 5C E9 0B 4E 46 FF ED 68 B9 DC A8
37 5D 92 86 E5 BA B4 3B 98 A7 BE 65 90 BF 84 83
```

x_0:

```
AE 67 0D 33 82 DF 4B 8D EC DE E0 B3 7D 2B FB A2
FD F4 C3 29 1B DB 74 F7 C1 CD B4 FD 63 41 C4 DE
A5 F7 8C 79 21 C4 5A 8B 54 63 9A 41 25 D3 1F 58
4E 82 56 A2 8D E0 1A 50 C2 96 A7 89 3E 07 33 61
0A 7D 99 BC 06 28 83 A5 A6 41 53 F9 CE 14 5D 71
0B 1E D6 5A 83 3D AB 44 ED 0F E0 65 3E 32 88 AF
BD 59 EE AC 85 8B FB DD F7 B8 4C 33 DD 5D A5 FE
A9 98 A9 D9 49 01 59 5B 40 C0 CE 5A 23 78 2A 48
```

r_0:

```
00 09 47 50 DB C7 43 16 75 05 8E 99 E5 2C 92 50
96 D9 CD 3E 81 57 E3 B8 F8 15 47 BB 49 A2 8F 50
27 18 3E BD 86 A3 36 21 5A 42 E8 03 AE 1B 62 27
55 55 9A D9 B7 FF 41 FD 83 4E 33 B2 E5 A2 B5 42
```

The correct value f_0 of the least significant byte of x_0 mod q is in this example $42h$ resp. 66 in decimal representation.

5.1 First Case: 8-Bit Architecture

The DPA calculation reveals the following list of the best 17 candidates (out of 256 candidates) for the correct value of f_0 on the base of 256 single measurements.

Besides to the correct value 66 (decimal notation) secondary positive correlation signals with decreasing amplitudes appear at the relative displacements of ± 128, ± 64, ± 32, ± 16, ± 8, ± 4, ± 2 and ± 1 of the correct hypothesis 66. If the number of single measurements is not a multiple of 256 the correlation coefficients of the secondary positive correlation coefficients differ slightly.

In the Fig. 2 both positive and negative correlation coefficients are taken into account. Negative correlation coefficients occur mainly at small correlation amplitudes. Strong correlation signals are caused by positive correlation coefficients.

5.2 Second Case: 32-Bit Architecture

During generation of the simulation data the 32-bit Hamming weight is used instead of an 8 bit Hamming weight. The DPA correlation is performed on the last 8 bits of the intermediate result after modular reduction.

Hypothesis	Correlation Coefficient	Relative Displacement of f_0
66	+1.000000	0
194	+0.750000	+128
2	+0.625000	-64
130	+0.625000	+64
34	+0.562500	-32
98	+0.562500	+32
50	+0.531250	-16
82	+0.531250	+16
58	+0.515625	-8
74	+0.515625	+8
62	+0.507812	-4
70	+0.507812	+4
64	+0.503906	-2
68	+0.503906	+2
65	+0.501953	-1
67	+0.501953	+1

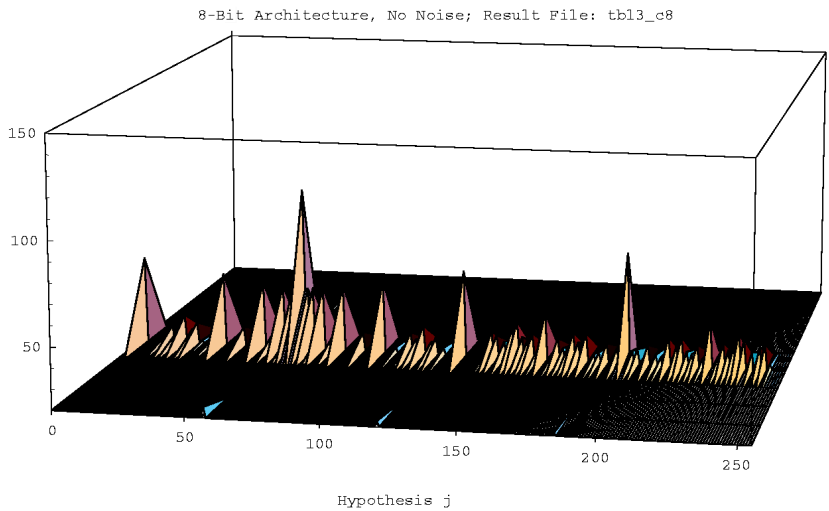

Fig. 1. Graphical representation of the absolute correlation coefficients on the base of 256 single measurements over time. Correlations coefficients $c(j,t) < |0.2|$ are neglected in this trace for clarity reasons.

The correlation coefficient at the correct value f_0 is the most significant and can be easily recognised (Fig. 3). It is more significant as in case of 32-bit random input values. Though the Hamming weight of the measurement series is based on 32 bit the correlation coefficients are nearly as significant as in case of an 8-bit architecture. Because of the equidistant step width only up to two bytes of input

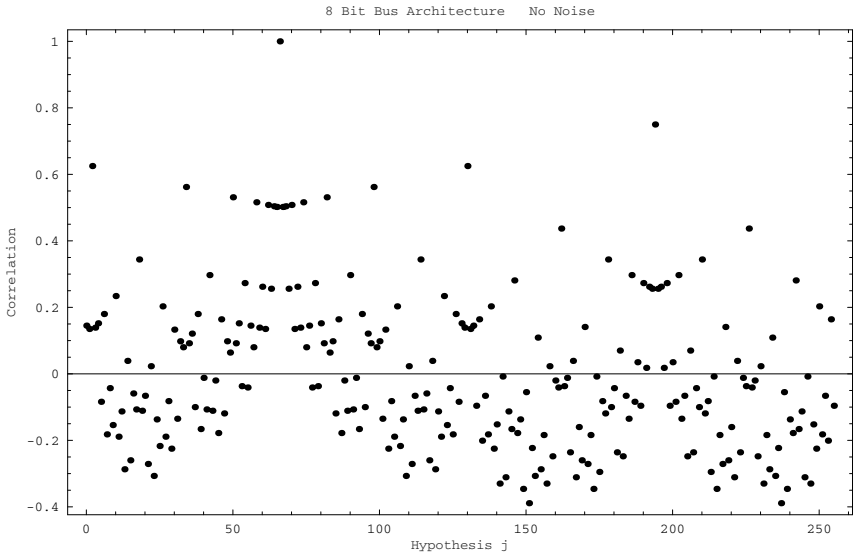

Fig. 2. Graphical representation of the correlation coefficients on the base of 256 single measurements. The smaller correlation amplitudes around $f_0 \pm 128$ of Fig. 1 turned out to be mainly of negative sign.

data and their contribution to the overall 32-bit Hamming weight are affected. A change at the more significant byte occurs at each 256th single measurement only.

It turned out that the DPA attack on the CRT is robust on different hardware architectures.

6 Efficiency of MRED

For the practical estimation of the attack potential we assume that we deal with 1024 bit RSA key size. The CRT implementation shall not include any restrictions on the input data and message blinding schemes that prevent MRED (see section 7). The smaller prime used is assumed to be about 500 bit long. After the first measurement series that serves as profiling step the further measurements series can be limited to a small time frame at the beginning of the CRT calculation that includes the modular reduction step. In general, it is assumed that an attacker needs a few hundred to a few thousand single measurements to prove DPA signals within one measurement series. The gcd check is successful after approximately 60-62 measurement series at the latest. For the overall number of single measurements we would expect $30.000 < n < 300.000$. The measurement itself consumes the most part of the time needed. For a rough estimation the overall measurement time t is expected to be 1 day $< t <$ 3 weeks. It further depends on the performance of the test set-up. The time of a DPA calculation

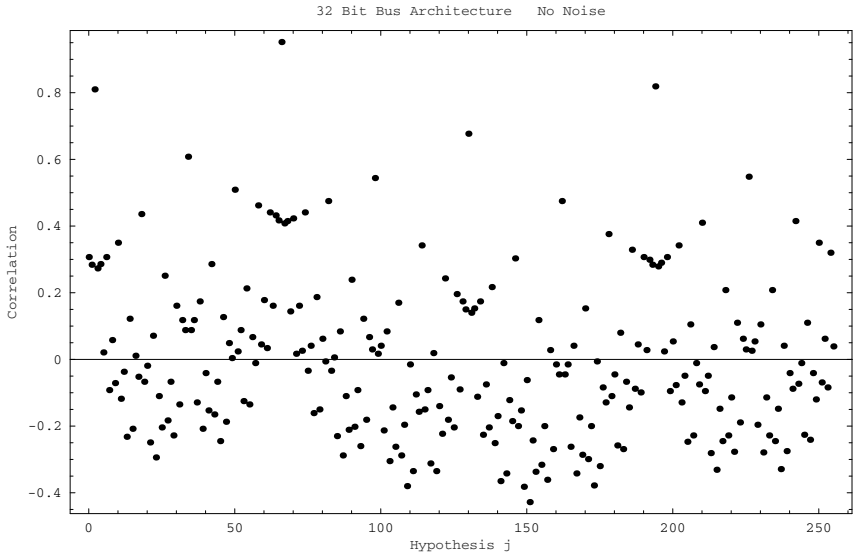

Fig. 3. Graphical representation of the correlation coefficients on the base of 256 single measurements. The correct value is $f_0 = 66$. The characteristics is shifted asymmetrically regarding to Fig. 2 (The correlation coefficient of $j = 2$ is nearly as high as of $j = 194$.)

on a standard PC should be done in the range of minutes depending on the time frame width that is used for the measurement. Additional time is generally necessary for re-synchronisation of single measurements. As it is possible to focus on a small time frame re-synchronisation should be done within minutes to at maximum 1 hour computing time for each measurement series.

MRED is linear to the bit size of the prime that is attacked. E. g. applying this attack to a 2048 bit RSA key will double the overall time.

Table 3. Summary of the Attack Efforts needed for a 1024 bit RSA key

Attack Tasks of MRED	
No. of Measurement Series:	60-62
No. of Single Measurements per Series:	500 - 5000
Single Measurement Data Size:	small
Overall Measurement Time:	1 day to 3 weeks
Overall Re-Synchronisation Time:	few hours to 2 days
No. of DPA calculations:	60-62
Overall DPA calculation time:	few hours to 1 day
Overall Time:	2 days to 1 month

Referring to the Common Criteria scheme [12] the attack potential of MRED is assessed to be in the range of 'Moderate' to 'High' for a 1024 bit RSA key. The assessment depends on the countermeasures of the implementation, the necessary adaptation work of the attacker and the further development of the attack methods.

7 Limitations and Countermeasures

There are three basic assumptions of the MRED attack, namely

1. a sufficient number of single measurements can be collected,
2. the input data x can be varied arbitrarily to construct equidistant input data, and
3. $(x_0 - i \cdot (256)^k) \bmod q$ holds $(r_0 - i \cdot (256)^k)$ at least for a subgroup of single measurements.

The first assumption deals with the number of single measurements that are needed for this DPA attack. As said before, this DPA attack against a 1024 bit RSA key demands for about $30.000 < n < 300.000$ single measurements. The upper boundary of single measurements may conflict with physical constraints of smart cards, e. g. if EEPROM write accesses are involved. Nevertheless, the authors assess that a few ten thousand measurements is a realistic number of exponentiations that can be carried out using a typical smart card. A general countermeasure to prevent these kind of statistical attacks is an usage counter for the number of RSA exponentiations. To secure the RSA decryption an additional failure counter can be implemented if a check of padding formats fails. On the other side an improvement of MRED may reduce the number of exponentiations.

The second assumption affects RSA signing ("chosen plaintext"), but not the RSA decryption ("chosen ciphertext"). The second assumption fails if the attacker has to deal with padding formats in case of digital signature applications. Typical padding formats used limit the range of variable data to the least significant 20 bytes of data that is the outcome of a hashing function. At the presence of padding formats MRED will reveal at maximum the least significant 20 bytes of the remainder of both primes p and q. Nevertheless, this information is of minor use as it doesn't give directly the least significant 20 bytes of the primes p and q, but of an unknown multiple of p and q, respectively. A possible way to proceed is a DPA attack that aims to find data correlations on the revealed bytes of the remainder during the following 'square - multiply' algorithm using the exponents d_p and d_q. If data correlations can be proven at the multiplications this leads to the disclosure of the exponents d_p and d_q. The occurence of these DPA signals during the exponentiation can be prevented by the common message blinding schemes.

The third assumption of MRED can be destroyed by message blinding. Multiplicative message blinding scheme as e. g. proposed by [11] use pairs (ν_i, ν_k) that are used for the blinding of the input data and unblinding of the result. This multiplicative blinding is applicable to prevent the likeliness of the third

assumption. Nevertheless, MRED might be useful to detect possible weaknesses within the message blinding scheme.

8 Conclusion

In this paper, we developed a new DPA attack on the remainder that can be applied at a CRT implementation of RSA to compromise one of the secret RSA primes. The basic assumption for MRED is that $(x_0 - i \cdot (256)^k) \bmod q$ holds $(r_0 - i \cdot (256)^k)$ at least for a subgroup of single measurements. The results of this MRED attack are shown on the base of simulated measurement data. Countermeasures against MRED should include the use of multiplicative blinding schemes to protect the reduction modulo a secret prime.

Acknowledgements. The authors would like to thank Robert Hammelrath and the team for the support of the work including the use of the DPA core routines and the test methods, as well as the referees for their valuable comments.

References

1. P. Kocher, J. Jaffe and B. Jun, "Differential Power Analysis", in: Proceedings of Advances in Cryptology – CRYPTO '99, Lecture Notes in Computer Science, Vol. 1666, Springer, Berlin 1999, pp. 388–397
2. T. S. Messerges, E. A. Dabbish and R. H. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", in: Proceedings of Workshop on Cryptographic Hardware and Embedded Systems, Springer, Lecture Notes in Computer Science, Vol. 1717, Springer, Berlin 1999, pp. 144–157
3. J. Kelsey, B. Schneier, D. Wagner, C. Hall, "Side Channel Cryptanalysis of Product Ciphers", in: Computer Security – ESORICS 98, Lecture Notes in Computer Science, Vol. 1485, Springer, Berlin 1998, pp. 487–496
4. T. S. Messerges, E. A. Dabbish, R. H. Sloan, "Investigations of Power Analysis Attacks on Smartcards", USENIX Workshop on Smartcard Technology, USENIX Association, 1999, pp. 151–161
5. J.-S. Coron, P. Kocher, D. Naccache, "Statistics and Secret Leakage", in: Financial Cryptography 2000, Lecture Notes in Computer Science, Vol. 1962, Springer, Berlin 2001, pp. 157–173
6. T. S. Messerges, "Using Second-Order Power Analysis to Attack DPA Resistant Software", in: Proceedings of Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, Vol. 1965, Springer, Berlin 2000, pp. 238–251
7. R. Mayer-Sommer, "Smartly Analyzing the Simplicity nd the Power of Simple Power Analysis on Smartcards", in: Proceedings of Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, Vol. 1965, Springer, Berlin 2000, pp. 78–92
8. P. N. Fahn and P. K. Pearson, "IPA: A New Class of Power Attacks", in Proceedings of Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, Vol. 1717, Springer, Berlin 1999, pp. 158–172

9. D. Boneh, R. A. DeMillo, R. J. Lipton, "On the Importance of Checking Cryptographic Protocols for Faults", in Advances in Cryptology – Eurocrypt 97, Lecture Notes in Computer Science, Vol. 1233, Springer, Berlin 1997, pp. 37–51
10. E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems", in: Advances in Cryptology – Crypto '97, Lecture Notes in Computer Science, Vol. 1294, Springer, Berlin 1997, pp. 513–525
11. P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Systems", in: Advances in Cryptology – Crypto '96, Lecture Notes in Computer Science, Vol. 1109, Springer, Berlin 1996, pp. 104–113
12. Common Criteria – Common Methodology for Information Technology Security Evaluation, CEM 99/045, Version 1.0, August 1999
13. A. J. Menezes, P. C. van Oorschot, S. C. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton 1997