On the Optimization of Side-Channel Attacks by Advanced Stochastic Methods

Werner Schindler

Bundesamt für Sicherheit in der Informationstechnik (BSI),
Godesberger Allee 185–189,
53175 Bonn, Germany
Werner.Schindler@bsi.bund.de

Abstract. A number of papers on side-channel attacks have been published where the side-channel information was not exploited in an optimal manner, which reduced their efficiency. A good understanding of the source and the true risk potential of an attack is necessary to rate the effectiveness of possible countermeasures. This paper explains a general approach to optimize the efficiency of side-channel attacks by advanced stochastic methods. The approach and its benefits are illustrated by examples.

Keywords: Side-channel attack, Montgomery's multiplication algorithm, stochastic process, statistical decision problem, optimal decision strategy.

1 Introduction

At Crypto 1996 and Crypto 1998 Kocher, resp. Kocher et al., introduced timing and power attacks [5, 8]. Since then side-channel attacks have attracted enourmous attention in the scientific community and the smart card industry as they constitute serious threats against cryptosystems. Their targets are usually smart cards but also software implementations may be vulnerable, even against remote attacks ([1, 2] etc.). In a side-channel attack the attacker guesses the secret key portion by portion. The correctness of the partial guesses cannot be verified (at least not with certainty) until all parts of the key have been guessed. If the verification of the whole key guess fails (e.g. by checking a digital signature) this does not provide the position(s) of the wrong guess(es).

A large number of research papers on timing attacks, power attacks, radiation attacks and combined timing / power attacks have been published. A variety of countermeasures have been proposed that shall prevent these attacks.

In 'real life' the number of measurements is often limited, or it is at least costly to perform a large number of measurements. From the attacker's point of view it is hence desirable to minimize the error probabilities for the guesses of the particular key parts (for a given number of measurements) or vice versa, to minimize the number of measurements which is necessary for a successful attack. If the outcome of the previous guesses has an impact on the guessing strategy of

S. Vaudenay (Ed.): PKC 2005, LNCS 3386, pp. 85–103, 2005.

the present key part it is additionally desirable to have criteria with which the correctness of the previous guesses can be verified with reasonable probability.

In order to achieve these goals the side-channel information should be exploited in an optimal manner. Many papers present ingenious ideas but lack of sound mathematical methods. As a consequence, only a fraction of the overall side-channel information is indeed used which in turn lowers the efficiency of the attack. As a consequence it may even be difficult to rate the true risk potential of these attacks and to assess the effectiveness of the proposed countermeasures. By applying appropriate stochastic methods it was possible to increase the efficiency of a number of known attacks considerably ([12, 14–16]; Sects. 4, 6, 7 in this paper), in one case even by factor 50. Moreover, some attacks were generalized, and new attacks were detected due to the better understanding of the situation ([13, 16, 17], Sects. 5, 6 in this paper).

The focus of this paper are the applied mathematical methods themselves but not new attacks. This shall put the reader into the position to apply and to adjust these methods when considering side-channel attacks that are tailored to a specific target. An individual treatment should in particular be necessary for most of the power and radiation attacks. Often the (timing, power, radiation) behaviour of the attacked device can be modelled as a stochastic process, and the attack can be interpreted as a sequence of statistical decision problems. Roughly speaking, in a statistical decision problem the optimal decision strategy minimizes the expected loss which primarily depends on the probabilities for wrong guesses but also on the a priori information and the consequences of errors. In fact, depending on the concrete situation particular types of errors may be easier to detect and correct than others (e.g., in the examples explained in Sects. 6 and 7).

We refer readers who are generally interested in stochastic and statistical applications in cryptography to ([10]) and to various papers of Meier and Staffelbach, Golic, Vaudenay and Junod, for instance.

Our paper is organized as follows: In Section 2 we introduce the concept of statistical decision theory, and in Section 3 we exemplarily work out a stochastic model for Montgomery's multiplication algorithm. Then we illustrate our approach by various examples and give final conclusions.

2 A Survey on Statistical Decision Theory

We interpret side-channel measurements as *r*ealizations of random variables, i.e. as values assumed by these random variables. The relevant part of the information is covered by noise but an attacker clearly aims to exploit all of the available information in an optimal way. Therefore, he interprets the side-channel attack as a sequence of statistical decision problems. Each decision problem corresponds to the guessing of a particular key part. Previous guesses may have an impact on the present guess (cf. Sects. 4, 5), or all guesses may be independent (cf. Sect. 6). Statistical decision theory quantifies the impact of the particular pieces of information on the decision so that the search for the optimal decision strategy

can be formalized. In this section we introduce the concept of statistical decision theory as far it is relevant for our purposes, namely to improve the efficiency of side-channel attacks.

Formally, a statistical decision problem is defined by a 5-tuple $(\Theta, \Omega, s, \mathcal{D}, A)$. The statistician (in our context: the attacker) observes a sample $\omega \in \Omega$ that he interprets as a realization of a random variable X with unknown distribution p_θ. On basis of this observation he estimates the parameter $\theta \in \Theta$ where Θ denotes the *p*arameter space, i.e., the set of all admissible hypotheses (= possible parameters). Further, the set Ω is called the *o*bservation space, and the letter A denotes the set of all admissible alternatives the statistician can decide for. In the following we assume $\Theta = A$ where Θ and A are finite sets.

Example 1. (i) Assume that the attacker guesses a single RSA key bit and that his decision is based upon N timing or power measurements. Then $\Theta = A = \{0, 1\}$, $\Omega = \mathbb{R}^N$.
(ii) Consider a power attack on a DES implementation where the attacker guesses a particular 6-bit-subkey that affects a single S-box in the first round. Then $\Theta = A = \{0, 1\}^6$.

A deterministic decision strategy is given by a mapping $\tau\colon \Omega \to A$ (cf. Remark 1 (ii)). If the statistician applies the decision strategy τ he decides for $\tau(\omega) \in A = \Theta$ whenever he observes $\omega \in \Omega$.

Finally, the *l*oss function $s\colon \Theta \times A \to [0, \infty)$ quantifies the harm of a wrong decision, i.e., $s(\theta, a)$ gives the loss if the statistician decides for $a \in A$ although $\theta \in \Theta = A$ is the correct parameter. In our context this quantifies the efforts (time, money etc.) to detect, to localize and to correct a wrong decision, i.e. a wrong guess of a key part. Clearly, $s(\theta, \theta) := 0$ since a correct guess does not cause any loss. For some attacks (as in Sects. 6, 7) specific types of errors are easier to correct than others. The optimal decision strategy takes such phenomena into account.

Assume that the statistician uses the deterministic decision strategy $\tau\colon \Omega \to A$ and that θ is the correct parameter. The expected loss (= average loss if the hypothesis θ is true) is given by the *r*isk function

$$r(\theta, \tau) := \int_\Omega s(\theta, \tau(\omega))\, p_\theta(d\omega). \tag{1}$$

Our goal clearly is to apply a decision strategy that minimizes this term. Unfortunately, usually there does not exist a decision strategy that is simultaneously optimal for all admissible parameters $\theta \in \Theta$. However, in the context of side-channel attacks one can usually determine (at least approximate) probabilities with which the particular parameters occur. This is quantified by the so-called *a* priori distribution η, a probability measure on the parameter space Θ.

Example 2. (i) (Continuation of Example 1(i)) Assume that k exponent bits remain to be guessed and that the attacker knows that r of them equal 1. If the secret key was selected randomly then it is reasonable to assume that the present bit is 1 with probability $\eta(1) = r/k$.
(ii) (Continuation of Example 1(ii)) Here $\eta(x) = 2^{-6}$ for all $x \in \{0, 1\}^6$.

Assume that η denotes the a priori distribution. If the statistician applies the deterministic decision strategy $\tau\colon \Omega \to A$ the expected loss equals

$$R(\eta,\tau) := \sum_{\theta\in\Theta} r(\theta,\tau)\eta(\theta) = \sum_{\theta\in\Theta} \int_{\Omega} s(\theta,\tau(\omega))\, p_\theta(d\omega)\, \eta(\theta). \tag{2}$$

A decision strategy τ' is *o*ptimal against η if it minimizes the right-hand term. Such a decision strategy is also called a *B*ayes strategy against η.

Remark 1. (i) For specific decision problems (e.g. minimax problems) it is reasonable to consider the more general class of *r*andomized decision strategies where the statistician decides randomly (quantified by a probability measure) between various alternatives when observing a particular ω ([20]). In our context we may restrict our attention to the deterministic decision strategies (cf. Theorem 1). We point out that deterministic decision strategies can be viewed as specific randomized decision strategies.
(ii) Theorem 1 is tailored to our situation. It provides concrete formulae that characterize the optimal decision strategy. Although Theorem 1 can be deduced from more general theorems (e.g., Hilfssatz 2.137 and Satz 2.138(i) in [20] immediately imply assertion (i)) we give an elementary proof (cf. Theorem 2.48 in [20] for the special case $t = 2$) in order to illustrate the background. We restricted our attention to the case $|\Theta| < \infty$ and left out mathematical difficulties as the concept of σ-algebras and measurability. We mention that the optimal decision strategy τ from Theorem 1 is measurable.

Theorem 1. *Assume that $(\Theta, \Omega, s, \mathcal{D}, A)$ describes a statistical decision problem with finite parameter space $\Theta = \{\theta_1, \ldots, \theta_t\} = A$ where $\mathcal{D}$ contains the deterministic decision strategies. Further, let μ denote a σ-finite measure on Ω with $p_{\theta_i} = f_{\theta_i} \cdot \mu$, i.e. p_{θ_i} has μ-density f_{θ_i}, for each $i \le t$.*
(i) The deterministic decision strategy $\tau\colon \Omega \to A$,

$$\tau(\omega) := a \quad \text{if } \sum_{i=1}^{t} s(\theta_i,a)\eta(\theta_i) f_{\theta_i}(\omega) = \min_{a'\in A} \left\{ \sum_{i=1}^{t} s(\theta_i,a')\eta(\theta_i) f_{\theta_i}(\omega) \right\} \tag{3}$$

is optimal against the a priori distribution η. (If the minimum is attained for several decisions, we chose $a \in A$ according to any (fixed) order on A.)
(ii) If $\Theta = \{0,1\}$ and $s(0,1), s(1,0) > 0$ the indicator function

$$\tau(\omega) := 1_{f_0(\omega)/f_1(\omega) \le s(1,0)\eta(1)/s(0,1)\eta(0)}(\omega) \tag{4}$$

is optimal against η. (We set $\tau(\omega) := 1$ if $f_0(\omega) = f_1(\omega) = 0$ or $f_0(\omega) = f_1(\omega) = \infty$.)
(iii) Assume that $C \subseteq \Omega$ with $p_{\theta_i}(C) = p > 0$ for all $\theta_i \in \Theta$. Then (i) and (ii) remain valid if f_θ is replaced by the conditional density $f_{\theta|C}$.

Proof. Let $\kappa\colon \Omega \times \mathcal{P}(A) \to [0,1]$ denote any randomized decision strategy (cf. [20], for instance). Fubini's Theorem implies

$$R(\eta,\kappa) = \sum_{i=1}^{t} \left(\int_{\Omega} \sum_{j=1}^{t} s(\theta_i,\theta_j)\kappa(\omega,\theta_j) f_{\theta_i}(\omega)\, \mu(d\omega) \right) \eta(\theta_i)$$

$$\int_{\Omega} \left(\sum_{i=1}^{t} \sum_{j=1}^{t} s(\theta_i, \theta_j) \kappa(\omega, \theta_j) f_{\theta_i}(\omega) \eta(\theta_i) \right) \mu(d\omega).$$

Since $\kappa(\omega, \cdot)$ is a probability measure, reordering the integrand yields

$$\sum_{j=1}^{t} \kappa(\omega, \theta_j) \sum_{i=1}^{t} s(\theta_i, \theta_j) f_{\theta_i}(\omega) \eta(\theta_i) \geq \min_{a' \in A} \left\{ \sum_{i=1}^{t} s(\theta_i, a') f_{\theta_i}(\omega) \eta(\theta_i) \right\}$$

which proves (3). Assertion (ii) is an immediate consequence from (i) since $f_0(\omega) = f_1(\omega) = 0$ and $f_0(\omega) = f_1(\omega) = \infty$ occur only with probability zero. Assertion (iii) is a corollary from (i) and (ii) since $f_{\theta|C} = f_\theta / p$.

Remark 2. (i) A σ-finite measure μ on Ω with the properties claimed in Theorem 1 does always exist (e.g. $\mu = p_{\theta_1} + \cdots + p_{\theta_t}$).
(ii) For $\Omega = \mathbb{R}^n$ the well-known Lebesgue measure λ_n is σ-finite (The Lebesgue measure on $\mathbb{R}^n$ is given by $\lambda_n([a_1, b_1] \times \cdots [a_n, b_n]) = \prod_{i=1}^{n} (b_i - a_i)$ if $b_i \geq a_i$ for all $i \leq n$.) If Ω is finite or countable the counting measure μ_C is σ-finite. The counting measure is given by $\mu_C(\omega) = 1$ for all $\omega \in \Omega$. In particular, the probabilities $\mathrm{Prob}_\theta(X = \omega) = p_\theta(\omega)$ can be interpreted as densities with respect to μ_C.
(iii) The examples mentioned in (ii) and combinations thereof cover the cases that are relevant in the context of side-channel attacks.

With regard to Theorem 1 we will restrict our attention to decision problems of the type

$$(\Theta, \Omega, s, \mathcal{DS}, A = \Theta) \quad \text{with finite } \Theta = A \tag{5}$$

where $\mathcal{DS}$ denotes the set of all deterministic decision strategies. At first the attacker has to define the sets $\Theta = A$ and an appropriate loss function s. Then he determines the a priori distribution η and, in particular, the probability densities p_{θ_i} for all $i \leq t$. In our context the latter will be the most difficult part but gives the most significant impact on the decision strategy. For timing attacks on public key algorithms, for instance, these distributions depend essentially on the implemented arithmetic algorithms, for power and radiation attacks on the internal activity within the attacked device or specific areas thereof when the measurements are taken. Finally, the attacker applies Theorem 1 to determine an optimal decision strategy τ (= Bayes strategy against the a priori distribution η). In specific situations the attacker may also be interested in the value $R(\eta, \tau)$.

3 Montgomery's Modular Multiplication Algorithm

In this section we investigate the timing behaviour of Montgomery's modular multiplication algorithm ([9], Alg. 14.36) as it is implemented in most of the smart cards that compute modular exponentiations (e.g., RSA-based digital signatures).

3.1 Algebraic Background and Montgomery's Algorithm

In this subsection we briefly describe the algebraic background and formulate the multiprecision variant of Montgomery's algorithm. We begin with a definition.

Definition 1. As usually, $Z_M := \{0, 1, \ldots, M-1\}$, and for an integer $b \in Z$ the term $b(\mathrm{mod}\, M)$ denotes the unique element of Z_M that is congruent to b modulo M.

In order to compute $y^d(\mathrm{mod}\, M)$ a sequence of modular multiplications and squarings have to be carried out. If 'ordinary' modular multiplication algorithms are used this requires a large number of time-consuming integer divisions by the modulus M. Montgomery's multiplication algorithm saves these operations.

In the following we assume that M is an odd modulus (e.g., an RSA modulus or a prime factor) and that $R := 2^x > M$ is a power of two (e.g. $x = 512$). The elementary variant of Montgomery's algorithm transfers the modular multiplications from the modulus Z_M to Z_R. The term $R^{-1} \in Z_M$ denotes the multiplicative inverse of R in Z_M, i.e. $RR^{-1} \equiv 1 \pmod{M}$. The integer $M^* \in Z_R$ satisfies the integer equation $RR^{-1} - MM^* = 1$. For input $a, b \in Z_M$ Montgomery's multiplication algorithm returns $\mathrm{MM}(a, b; M) := abR^{-1}(\mathrm{mod}\, M)$. We point out that the mappings $\Psi, \Psi_*: Z_M \to Z_M$, given by $\Psi(x) := xR\,(\mathrm{mod}\, M)$ and $\Psi_*(x) := xR^{-1}\,(\mathrm{mod}\, M)$, induce inverse operations on Z_M.

Usually, a time-efficient multiprecision variant of Montgomery's algorithm is implemented which is tailored to the device's hardware architecture. Assume that ws denotes the word size for the arithmetic operations (e.g. $ws = 32$) and that ws divides the exponent x. Then $r := 2^{ws}$ and $R = r^v$ with $v = x/ws$ (Example: $x = 512$, $ws = 32$, $v = 16$). For the moment let further $a = (a_{v-1}, \ldots, a_0)_r$, $b = (b_{v-1}, \ldots, b_0)_r$, and $s = (s_{v-1}, \ldots, s_0)_r$ denote the r-adic representations of a, b and s, resp., and let $m' := M^*(\mathrm{mod}\, r)$.

Algorithm 1: Montgomery's Algorithm (Multiprecision Variant)

```
1.) s:=0
2.) for i=0 to v-1 do {
      u_i:= (s_0+a_i*b_0)m' (mod r)
      s:= (s+a_ib+u_iM) /r }
3.) if s≥M then s:=s-M
4.) return s  ( = MM(a,b;M) = abR^{-1} (mod M) )
```

In the following we assume that for fixed parameters M, R and r the run times needed for Step 1 and Step 2 are identical for all pairs of operands. (This assumption is reasonable, in particular for smart cards. Software implementations may process small operands (i.e., those with leading zero-words) faster due to optimizations of the integer multiplication algorithms. This is absolutely negligible for the attacks considered in Sects. 4 and 6 but may cause additional difficulties for particular chosen-input attacks as described in Sect. 5, for instance (cf. [1]).) Timing differences are caused by the fact whether in Step 3 the subtraction, the so-called *extra reduction*, has to be carried out. Hence

$$\mathrm{Time}\,(MM(a, b; M)) \in \{c, c + c_{\mathrm{ER}}\} \tag{6}$$

where the time c is required iff no extra reduction is necessary. The constant c_{ER} quantifies the time needed for an integer subtraction by M. The values of the constants c and c_{ER} surely depend on the concrete implementation. Lemma 1 below (cf. [13] (Remark 1) or [11] (Lemma 1)) says that the fact whether an extra reduction is necessary does only depend on a, b, M and R but not on the word size ws.

Lemma 1. *For each word size ws the intermediate result after Step 2 equals $s = (ab + uM)/R$ with $u = abM^* (\mathrm{mod}\, R)$.*

3.2 The Stochastic Model

In this subsection we study the timing behaviour of Montgomery's multiplication algorithm within modular exponentiation algorithms. It will turn out that the probability for an extra reduction (ER) in a squaring operation differs from the probabiliy for an extra reduction in a multiplication with a particular value $a \in Z_M$. The latter depends linearly on the ratio a/M. We point out that the probabilities, or more general, the stochastic properties of random extra reductions do not depend on the size of the modulus M but on the ratio M/R.

Lemma 2. *(i)* $\frac{\mathrm{MM}(a,b;M)}{M} = \left(\frac{a}{M}\frac{b}{M}\frac{M}{R} + \frac{abM^* \pmod R}{R}\right) (\mathrm{mod}\, 1)$. *That is, an extra reduction is carried out iff the sum within the bracket is ≥ 1 iff* $\frac{\mathrm{MM}(a,b;M)}{M} < \frac{a}{M}\frac{b}{M}\frac{M}{R}$.
(ii) Assume that the random variable B is equidistributed on Z_M. Then the intermediate result in Algorithm 1 before the ER step is (in good approximation) distributed as

$$\frac{M}{R}\frac{a}{M}U + V \qquad \textit{for } \mathrm{MM}(a, B; M) \tag{7}$$

$$\frac{M}{R}U^2 + V \qquad \textit{for } \mathrm{MM}(B, B; M). \tag{8}$$

where U and V denote independent random variables that are equidistributed on $[0, 1)$.

Sketch of the Proof. Assertion (i) follows immediately from Lemma 1. For a proof of (ii) we refer the interested reader to [12], Lemma A.3. The central idea is that a small deviation of B/M causes 'vast' deviations in the second summand and that the distribution of the second summand is close to the equidistribution on $[0, 1]$ for nearly all values of a. An alternate proof for a related assertion is given in [11]. (Both proofs use plausible heuristic arguments (Assumption DIS in [11]). We further mention that (7) and (8) yield probabilities for extra reductions (cf. (11)) which were confirmed by a large number of simulation experiments.

Modular exponentiation algorithms initialize a variable (in the following denoted with temp) with the base y or, in case of table methods, with a power of y. A sequence of modular squarings of temp and multiplications of temp

with particular table values are carried out until temp equals $y^d (\bmod M)$. Pseudoalgorithm 2 below combines modular exponentiation algorithms with Montgomery's multiplication algorithm. The modular exponentiation algorithm may be the 'square and multiply' algorithm ([9], Alg. 14.79; cf. Sect. 4), a table method (e.g. left-to right b-ary exponentiation, cf. [9], Alg. 14.82 and Sect. 6) or the sliding windows exponentiation algorithm ([9], Alg. 14.85). In Pseudoalgorithm 2 the table values equal $(y^j R) \,(\mathrm{mod}\, M)$ (unlike $(y^j) (\mathrm{mod}\, M)$ if 'ordinary' modular multiplication algorithms are used) and hence temp $= y^d R (\mathrm{mod}\, M)$ after Step 2.

Pseudoalgorithm 2: Modular Exponentiation with Montgomery's Multiplication Algorithm

```
1.) \bar y_{1}:=MM(y,R^2;M)   (= yR (mod M))
2.) Modular Exponentiation algorithm
    a) table initialization (if necessary)
    b) exponentiation phase
       (Replace modular squarings and multiplications in
       2a) and 2b) with the respective Montgomery operations)
3.) return temp:=MM(temp,1;M)   (=y^d (mod M))
```

We interpret the normalized intermediate values $\mathrm{temp}_0/M, \mathrm{temp}_1/M, \ldots$ from the exponentiation phase as realizations of $[0,1)$-valued random variables $S_0, S_1, \ldots$. Consequently, the time needed for the i^{th} Montgomery operation (squaring or multiplication of temp with a particular table value), is interpreted as a realization of $c + W_i \cdot c_{\mathrm{ER}}$, where W_i is a $\{0,1\}$-valued random variable, assuming 1 iff an extra reduction is necessary. The understanding of the stochastic process $W_1, W_2, \ldots$ will turn out to be necessary to determine the optimal decision strategies in the following sections.

From Lemma 2 we deduce the following relations where the right-hand sides denote the possible types of the i^{th} Montgomery operation within the exponentiation phase.

$$S_{i+1} := \begin{cases} \frac{M}{R} S_i^2 + V_{i+1} (\mathrm{mod}\, 1) & \text{for MM(temp, temp; } M) \\ \frac{\bar y_j}{M} \frac{M}{R} S_i + V_{i+1} (\mathrm{mod}\, 1) & \text{for MM(temp, } \bar y_j; M) \end{cases} \tag{9}$$

The term $\bar y_j$ denotes the j^{th} table entry ($j \equiv 1$ for the square & multiply algorithm). With regard to Lemma 2(ii) we may assume that the random variables $V_1, V_2, \ldots$ are iid equidistributed on $[0,1)$. As an immediate consequence, the random variables $S_1, S_2, \ldots$ are also iid equidistributed on $[0,1)$. From the random variables $S_0, S_1, \ldots$ one derives the random variables $W_1, W_2, \ldots$ that describe the (random) timing behaviour of the Montgomery operations within the exponentiation phase. To be precise, from Lemma 2(i) we conclude

$$W_i := \begin{cases} 1_{S_i < S_{i-1}^2 (M/R)} & \text{for MM(temp, temp; } M) \\ 1_{S_i < S_{i-1} (\bar y_j / M)(M/R)} & \text{for MM(temp, } \bar y_j; M). \end{cases} \tag{10}$$

We mention that the sequence $W_1, W_2, \ldots$ is neither independent nor identically distributed but W_i and W_{i+1} are negatively correlated. On the other hand, the

tuples $(W_i, W_{i+1}, \ldots, W_{i+j})$ and $(W_k, W_{k+1}, \ldots, W_{k+t})$ (but not their components!) are independent if $k > i + j + 1$. In particular, (10) implies

$$E(W_i) = \begin{cases} \frac{1}{3}\frac{M}{R} & \text{for MM(temp, temp; } M) \\ \frac{1}{2}\frac{\bar{y}_j}{M}\frac{M}{R} & \text{for MM(temp, } \bar{y}_j; M). \end{cases} \quad (11)$$

Remark 3. In this section we have derived a stochastic process $W_1, W_2, \ldots$ that models the timing behaviour of the Montgomery multiplications within modular exponentiation algorithms. Clearly, a similar approach is at least principally feasible for other arithmetic algorithms, too.

4 Timing Attacks on RSA Without CRT

A timing attack on RSA implementations was first described (and experimentally verified) in [5]. Two years later a successful timing attack on a preliminary version of the Cascade chip was presented at the Cardis conference ([4]). Kocher's attack was generalized and optimized in [12]. In this section we consider the attack presented in [4]. Our approach improves its efficiency by factor 50.

4.1 The Optimal Decision Strategy

In this section we assume that the attacked smart card (e.g., a preliminary version of the Cascade chip) calculates the modular exponentiations $y \mapsto y^d (\mathrm{mod}\ n)$ with the square & multiply algorithm, combined with Montgomery's algorithm (cf. Pseudoalgorithm 2). We assume further that the secret exponent d (target of the attack) remains fixed for all observed exponentiations and that no blinding techniques are applied (cf. Remark 4) so that repetitions with identical bases require equal running times. The binary representation of the secret exponent d reads $(d_{v-1}, \ldots, d_0)_2$, and in Phase 2b of Pseudoalgorithm 2 the exponent bits are processed from the left to the right.

In a pre-step the attacker measures the exponentiation times $\widetilde{t}_{(j)} := \mathrm{Time}(y_{(j)}^d (\mathrm{mod} n)) + t_{\mathrm{Err(j)}}$ for a sample $y_{(1)}, \ldots, y_{(N)}$ where $t_{\mathrm{Err(j)}}$ denotes the measurement error for sample j. To be precise, we have

$$\widetilde{t}_{(j)} = t_{\mathrm{Err}(j)} + t_{S(j)} + (v + \mathrm{ham}(d) - 2)c + \left(w_{(j)1} + \ldots + w_{(j)v+\mathrm{ham}(d)-2}\right) c_{\mathrm{ER}} \quad (12)$$

where $w_{(j)i} \in \{0, 1\}$ equals 1 iff the i^{th} Montgomery operation requires an extra reduction for sample j and 0 else. The term $t_{S(j)}$ summarizes the time needed for all operations apart from the Montgomery multiplications (input, output, handling the loop variable, evaluating the if-statements, pre- and post-multiplication). We may assume that the attacker knows $t_{S(j)}$ exactly as possible errors can be interpreted as part of the measurement error $t_{\mathrm{Err}(j)}$. We may further assume that the attacker had guessed the parameters v, $\mathrm{ham}(d)$, c and c_{ER} in a pre-step of the attack (cf. [14], Sect. 6).

The exponent bits are guessed from the left to the right. For the moment we assume that the most significant exponent bits $d_{v-1}, \ldots, d_{k+1}$ have already

been guessed, and that all guesses $\widetilde{d}_{v-1}, \ldots, \widetilde{d}_{k+1}$ are correct. Our goal is to derive an optimal decision strategy to guess the exponent bit d_k. At first the attacker subtracts the time needed to process the (correctly) guessed exponent bits $d_{v-1}, \ldots, d_{k+1}$ from the measured exponentiation time in order to obtain the time needed for the remaining bits $d_k, \ldots, d_0$ (beginning with 'if $(d_k = 1)$ then $\mathrm{MM}(\mathrm{temp}_{(j)}, \bar{y}_{1(j)}; n)$'), and from $\mathrm{ham}(d)$ he further computes the number m of remaining Montgomery multiplications with $\bar{y}_{1(j)}$. If the random exponent d has been selected randomly it is reasonable to assume that $\eta(1) := \mathrm{Prob}(d_k = 1) = (m-1)/k$ since $d_0 = 1$. That is, the a priori distribution is given by $(\eta(0), \eta(1)) = ((k+1-m)/k, (m-1)/k)$. Clearly, $\Theta = A = \{0,1\}$. Since the differences of the running times are caused by the number of extra reductions (and maybe by measurement errors) we consider the 'normalized' remaining time

$$\widetilde{t}_{d,rem(j)} := \frac{\widetilde{t}_{(j)} - t_{S(j)} - (v + \mathrm{ham}(d) - 2)c}{c_{\mathrm{ER}}} - \sum_{i=1}^{v+\mathrm{ham}(d)-2-k-m} w_{i(j)} \quad (13)$$

$$= t_{dErr(j)} + \sum_{i=v+\mathrm{ham}(d)-k-m-1}^{v+\mathrm{ham}(d)-2} w_{i(j)}.$$

where the last sum equals the number of extra reductions in the remaining Mongomery multiplications. The remaining Montgomery operations are labelled by the indices $v + \mathrm{ham}(d) - k - m - 1, \ldots, v + \mathrm{ham}(d) - 2$. The normalized measurement error $t_{dErr(i)} = t_{Err(i)}/\, c_{\mathrm{ER}}$ is assumed to be a realization of an $N(0, \alpha^2 (= \sigma^2_{Err}/\, {c_{\mathrm{ER}}}^2))$-distributed random variable that is independent of $W_1, W_2, \ldots$ (cf. [12], Sect. 6).

The attacker bases his decision on the $4N$-tuple

$$\left(\widetilde{t}_{\mathrm{drem}(j)}, u_{M(j)}, u_{S(j)}, t_{S(j)}\right)_{j \le N}$$

('observation') where $u_{M(j)}, u_{S(j)}, t_{S(j)} \in \{0,1\}$ quantify the timing of sample j until the next decision (i.e., when guessing d_{k-1}). To be precise, $u_{M(j)} = 1$ (resp. $u_{S(j)} = 1$, resp. $t_{S(j)} = 1$) iff $\theta = 1$ and the next multiplication with $\bar{y}_{1(j)}$ (resp., iff $\theta = 1$ and the subsequent squaring, resp. iff $\theta = 0$ and the next squaring) requires an extra reduction. That is, $u_{M(j)}$ and $u_{S(j)}$ are summands of the right-hand side of (13) if $\theta = 1$ whereas $t_{S(j)}$ is such a summand if $\theta = 0$. Next, we study the stochastic process $W_{1(j)}, W_{2(j)}, \ldots$ that quantifies the (random) timing behaviour of these Montgomery multiplications. Although these random variables are neither independent nor stationary distributed they yet meet the central limit theorem ([12], Lemma 6.3(iii)). Since $W_{i(j)}$ and $W_{r(j)}$ are independent if $|i - r| > 1$ (cf. Subsection 3.2) we conclude

$$\mathrm{Var}\left(W_{1(j)} + \ldots + W_{t(j)}\right) = \sum_{i=1}^{t} \mathrm{Var}(W_{i(j)}) + 2 \sum_{i=1}^{t-1} \mathrm{Cov}(W_{i(j)}, W_{i+1(j)}) \quad (14)$$

Concerning the variances we have to distinguish between two cases (squaring, multiplication with $\bar{y}_{(j)}$; cf. (11)), for the covariances between three cases, namely

that $W_{i(j)}$ and $W_{i+1(j)}$ correspond to two squarings ($\mathrm{cov_{SS}}$), resp. to a squaring followed by a multiplication with $\bar{y}_{(j)}$ ($\mathrm{cov_{SM(j)}}$), resp. to a multiplication with $\bar{y}_{(j)}$ followed by a squaring ($\mathrm{cov_{MS(j)}}$). Exploiting (10) and (9) the random vector $(W_{i(j)}, W_{i+1(j)})$ can be expressed as a function of the iid random variables $S_{i-1(j)}, S_{i(j)}, S_{i+1(j)}$. For instance, $\mathrm{Cov_{MS}}(W_i W_{i+1}) =$

$$\int_{[0,1)^3} 1_{\{s_i < s_{i-1}\bar{y}_j/R\}} \cdot 1_{\{s_{i+1} < s_i^2 n/R\}}(s_{i-1}, s_i, s_{i+1})\, ds_{i-1} ds_i ds_{i+1} - \frac{\bar{y}_{(j)}}{2R} \cdot \frac{n}{3R} \quad (15)$$

Careful but elementary computations yield

$$\mathrm{cov_{MS(j)}} = 2p_j^3 p_* - p_j p_*, \quad \mathrm{cov_{SM(j)}} = \frac{9}{5} p_j p_*^2 - p_j p_* \quad (16)$$

$$\mathrm{cov_{SS}} = \frac{27}{7} p_*^4 - p_*^2 \qquad \text{with } p_j := \frac{\bar{y}_{(j)}}{2R} \text{ and } p_* := \frac{n}{3R}.$$

Since the random variables $W_{1(j)}, W_{2(j)}, \ldots$ are not independent the distribution of $W_{i+1(j)} + \cdots + W_{t(j)}$ depends on the preceding value $w_{i(j)}$. Theorem 2 considers this fact (cf. [12]). We first introduce some abbreviations.

Notation. $hn(0,j) := (k-1)p_*(1-p_*) + mp_j(1-p_j) + 2(m-1)\mathrm{cov_{MS(j)}} + 2(m-1)\mathrm{cov_{SM(j)}} + 2(k-m-1)\mathrm{cov_{SS}} + 2\frac{k-m}{k-1}\mathrm{cov_{SM(j)}} + 2\frac{m-1}{k-1}\mathrm{cov_{SS}} + \alpha^2$,
$hn(1,j) := (k-1)p_*(1-p_*) + (m-1)p_j(1-p_j) + 2(m-2)\mathrm{cov_{MS(j)}} + 2(m-2)\mathrm{cov_{SM(j)}} + 2(k-m)\mathrm{cov_{SS}} + 2\frac{k-m+1}{k-1}\mathrm{cov_{SM(j)}} + 2\frac{m-2}{k-1}\mathrm{cov_{SS}} + \alpha^2$,
$ew(0,j \mid b) := (k-1)p_* + mp_j + \frac{k-m}{k-1}(p_{*S(b)} - p_*) + \frac{m-1}{k-1}(p_{jS(b)} - p_j)$,
$ew(1,j \mid b) := (k-1)p_* + (m-1)p_j + \frac{k-m+1}{k-1}(p_{*S(b)} - p_*) + \frac{m-2}{k-1}(p_{jS(b)} - p_j)$ with $p_{*S(1)} := \frac{27}{7}p_*^3$, $p_{*S(0)} := \frac{p_* - p_* p_{*S(1)}}{1-p_*}$, $p_{jS(1)} := \frac{9}{5}p_* p_j$ and $p_{jS(0)} := \frac{p_j - p_* p_{jS(1)}}{1-p_*}$.

A false guess $\widetilde{d}_k \neq d_k$ implies wrong assumptions about the intermediate temp values for both hypotheses $d_t = 0$ and $d_t = 1$ for all the forthcoming decisions (when guessing d_t for $t < k$). Consequently, these guesses cannot be reliable, and hence we use the loss function $s(0,1) = s(1,0) = 1$. (For this setting the expected loss $R(\eta, \tau)$ equals the error probability $\mathrm{Prob}(d_k \neq \widetilde{d}_k)$.) For a complete proof of Theorem 2 we refer the interested reader to [12], Theorem 6.5 (i).

Theorem 2. *(Optimal decision strategy) Assume that the guesses $\widetilde{d}_{v-1}, \ldots, \widetilde{d}_{k+1}$ are correct and that* $\mathrm{ham}(d_k, \ldots, d_0) = m$. *Let*

$$\psi_{N,d} : (\mathbb{R} \times \{0,1\}^3)^{\mathrm{N}} \to \mathbb{R}, \qquad \psi_{\mathrm{N,d}}((\widetilde{\mathrm{t}}_{\mathrm{drem(1)}}, \mathrm{u_{M(1)}}, \ldots, \mathrm{u_{S(N)}}, \mathrm{t_{S(N)}})) :=$$
$$-\frac{1}{2}\sum_{j=1}^{N}\left(\frac{\left(\widetilde{t}_{\mathrm{drem}(j)} - t_{S(j)} - ew(0,j \mid t_{S(j)})\right)^2}{hn(0,j)} - \frac{\left(\widetilde{t}_{\mathrm{drem}(j)} - u_{M(j)} - u_{S(j)} - ew(1,j \mid u_{S(j)})\right)^2}{hn(1,j)}\right).$$

Then the deterministic decision strategy τ_d: $(\mathbb{R} \times \{0,1\}^3)^{\mathrm{N}} \to \{0,1\}$, defined by

$$\tau_d = 1_{\psi_{N,d} < \log(\frac{m-1}{k-m+1}) + \frac{1}{2}\sum_{j=1}^{N}\log(1+c_j)} \quad \text{with } c_j := \frac{hn(0,j) - hn(1,j)}{hn(1,j)} \quad (17)$$

is optimal (i.e., a Bayes strategy against the a priori distribution η).

Sketch of the Proof. To apply Theorem 1(ii), (iii) we first have to determine the conditional probability densities $h_{\theta,*j|C_j}(\widetilde{t}_{\mathrm{drem}(j)}, u_{M(j)}, u_{S(j)}, t_{S(j)})$ (normal distribution) of the random vectors $\boldsymbol{X}_j := (\widetilde{T}_{\mathrm{drem}(j)}, U_{M(j)}, U_{S(j)}, T_{S(j)})$ for $\theta = 0, 1$ and $j \le N$ with $C_j = (U_{M(j)} = u_{M(j)}, U_{S(j)} = u_{S(j)}, T_{S(j)} = t_{S(j)})$. (We point out that the $\boldsymbol{X}_j$ are independent but not their components.) The products $\prod_{j=1}^{N} h_{\theta,*j|C_j}(\cdot)$ are inserted in (4), and elementary computations complete the proof of Theorem 2.

The overall attack is successful iff all the guesses $\widetilde{d}_{v-1}, \ldots, \widetilde{d}_0$ are correct. Theorem 6.5 (ii) in [12] quantifies the probability for individual wrong guesses. In particular, guessing errors will presumably only occur in the first phase of the attack since the variance of the sum $W_{v+\mathrm{ham}(d)-k-m-1(j)} + \ldots + W_{v+\mathrm{ham}(d)-2(j)}$ decreases as k tends to 0. Due to the lack of space we skip this aspect but give a numerical example.

Example 3. Assume that the guesses $\widetilde{d}_{v-1}, \ldots, \widetilde{d}_{k+1}$ have been correct. For randomly chosen bases $y_{(1)}, \ldots, y_{(N)}$, for $n/R = 0.7$, $\alpha^2 = 0$, $N \ge 5000$, and ...
(a) ... $(k, m) = (510, 255)$ we have $\mathrm{Prob}(\widetilde{d}_k \ne d_k) \le 0.014$.
(b) ... $(k, m) = (440, 234)$ we have $\mathrm{Prob}(\widetilde{d}_k \ne d_k) \le 0.010$.
(c) ... $(k, m) = (256, 127)$ we have $\mathrm{Prob}(\widetilde{d}_k \ne d_k) \le 0.001$.

4.2 Error Detection, Error Location and Error Correction

In order to guess the secret exponent d the attacker considers a sequence of statistical decision problems (one for each exponent bit). The $\psi_{N,d}$-values themselves can be interpreted as realizations of random variables $Z_{v-1}, Z_{v-2}, \ldots$ with the pleasant property that their distributions change noticeably after the first wrong guess. For instance, the decision strategy from Theorem 2 then yields the guess 1 only with a probability of about 0.20 (The exact probability depends on the concrete parameters; cf. [12], Theorem 6.5(iii)). The interested reader is referred to Section 3 of [15] where a new stochastic strategy was introduced to detect, locate and correct guessing errors, which additionally reduces the sample size by about 40%.

4.3 Practical Experiments/Efficiency of the Optimized Attack

Reference [15] distinguishes two cases. In the *i*deal case it is assumed that the time measurements are exact, that the attacker knows the constants and parameters c, c_{ER}, v and $\mathrm{ham}(d)$ and that he is able to determine the setup time $t_{(S)}$ exactly. For the 'real-life' case the timing measurements were performed using an emulator which predicts the running time of a program in clock cycles. The code we used was the ready-for-transfer version of the Cascade library, i.e. with critical routines directly written in the card's native assemble language. Since the emulator is designed to allow implementors to optimize their code before 'burning' the actual smart cards, its predictions should match almost perfectly. In the 'real-life' case the attacker did not know $c, c_{\mathrm{ER}}, v, \mathrm{ham}(d)$, and $t_{(S)}$. Instead, these values were guessed in a pre-step ([14], Sect. 6).

Applying the optimized decision strategy and the error detection strategy mentioned in the previous subsection we obtained for sample size $N = 5000$ success rates of 85% (ideal case) and 74% ('real-life' case). For $N = 6000$ we obtained success rates of 95% and 85%, respectively. The original attack ([4]) yet required $200.000 - 300.000$ measurements. In other words: The optimized decision strategy from Theorem 2, combined with an efficient new error detection strategy, improved the efficiency of the original attack by factor 50. Moreover, the success rates for the ideal and the 'real-life' case are of the same size, which additionally underlines that our stochastic model is very appropriate.

Remark 4. (Countermeasures). The attacker exploits that the secret exponent d is the same for each exponentiation and that he knows both the bases and the modulus. In fact, this type of timing attack can be prevented with exponent blinding or base blinding techniques ([5]; Sect. 10). The latter is yet not sufficient to prevent combined timing and power attacks (cf. Sect. 6). Constant processing times for all Montgomery operations clearly is an alternative countermeasure. This goal can be reached by omitting all extra reductions within the exponentiation phase at cost of a larger modulus $R > 4M$ ([18]). Alternatively, an integer subtraction may be carried out in each Montgomery operation. (The dummy subtractions should be implemented carefully since otherwise the compiler might ignore them.)

5 A Timing Attack on RSA with CRT

In the previous section we considered a timing attack on RSA implementations that do not use the CRT. It was essential that the attacker knew the base y, the modulus n and the intermediate results of the computation. These requirements are obviously not fulfilled if the CRT is used. Consequently, it had been assumed for some years that CRT implemenations were not vulnerable to timing attacks. In [13] a new type of timing attack against RSA with CRT and Montgomery's multiplication algorithm was introduced (adaptive chosen-input attack). Unlike the attack from the previous section it does not guess the secret exponent d bit by bit but factorizes the modulus $n = p_1 p_2$. The attack would not have been detected without the understanding of the stochastic behaviour of Montgomery's multiplication algorithm. We point out that also this timing attack can be prevented with the countermeasures mentioned in Remark 4.

If the CRT is applied $x_i := (y(\mathrm{mod}\, p_i))^{d_i} \equiv y^d(\mathrm{mod}\, p_i)$ is computed for $i = 1, 2$ with $d_i = d(\mathrm{mod}\,(p_i - 1))$. Finally, $y^d(\mathrm{mod}\, n)$ is computed from these intermediate results. We assume that the square & multiply exponentiation algorithm and Montgomery's algorithm are used to calculate x_1 and x_2. As in the previous section $R > p_i$ denotes the Montgomery constant (which is assumed to be the same for p_1 and p_2), while R^{-1} stands for the multiplicative inverse of R in $\mathrm{Z_n}$. For input $y := uR^{-1}(\mathrm{mod}\, n)$ the constant factor in the computation of x_i equals $\bar{y}_{i;1} = yR \equiv u\,(\mathrm{mod}\, p_i)$ (cf. Step 1 of Pseudoalgorithm 2).

Let $0 < u_1 < u_2 < n$ with $u_2 - u_1 \ll p_1, p_2$. Three cases are possible: The 'interval set' $\{u_1 + 1, \ldots, u_2\}$ contains no multiple of p_1 and p_2 (Case A),

resp. contains a multiple of p_1 or p_2 but not of both (Case B), resp. contains multiples of both p_1 and p_2 (Case C). The computation of x_i requires about $\log_2(n)/2$ squarings and $\log_2(n)/4$ multiplications with $\bar{y}_{i;1}$. The running time for input $y := uR^{-1} \pmod n$, denoted with $T(u)$, is interpreted as a realization of a normally distributed random variable X_u (cf. [13]), and from (11) we obtain

$$E(X_{u_2} - X_{u_1}) \approx \begin{cases} 0 & \text{for Case A} \\ -\frac{c_{\mathrm{ER}}}{8}\frac{\sqrt{n}}{R} & \text{for Case B} \\ -\frac{c_{\mathrm{ER}}}{4}\frac{\sqrt{n}}{R} & \text{for Case C.} \end{cases} \tag{18}$$

where '$E(\cdot)$' denotes the expectation of a random variable. This observation can be used for a timing attack that factorizes the modulus n. In Phase 1 the attacker determines an 'interval set' $\{u_1 + 1, \ldots, u_2\}$ with $u_2 - u_1 \approx 2^{-6}p_1, 2^{-6}p_2$ that contains a multiple of p_1 or p_2. The attacker is convinced that this is the case iff $T(u_2) - T(u_1) > -c_{\mathrm{ER}}\sqrt{n}/16R$. (There is no need to distinguish between Case B and Case C.) Starting with this interval $\{u_1+1, \ldots, u_2\}$ in Phase 2 he applies the same decision rule to decide whether its upper halve contains a multiple of p_1 or p_2, and he replaces current interval by that halve (upper halve or lower halve) that contains a multiple of p_1 or p_2. In the elementary form of the attack this process is continued until the actual subset $\{u_1 + 1, \ldots, u_2\}$ is sufficiently small so that it is feasible to calculate $\gcd(u, n)$ for all u within this subset (Phase 3). If all decisions within Phase 1 and Phase 2 have been correct the final subset indeed contains a multiple of p_1 or p_2, and Phase 3 yields the factorization of n.

At any instant within Phase 2 the attacker can verify with high probability whether his decisions have been correct so far, i.e. whether a given interval $\{u_1+1, \ldots, u_2\}$ really contains a multiple of p_1 or p_2. He just applies the decision rule to the time difference required for neighboured values of u_1 and u_2, for instance to $T(u_2 - 1) - T(u_1 + 1)$. If this confirms the preceding decisions it is verified with overwhelming probability that the interval $\{u_1 + 1, \ldots, u_2\}$ truly contains a multiple of p_1 or p_2. Consequently, we then call $\{u_1 + 1, \ldots, u_2\}$ a *confirmed interval*. Otherwise, the attacker evaluates a further time difference (e.g. $T(u_2-2)-T(u_1+2)$). Depending on this difference he either finally confirms the interval $\{u_1 + 1, \ldots, u_2\}$ or restarts the attack at the preceding confirmed interval, denoted with $\{u_{1;c} + 1, \ldots, u_{2;c}\}$, using values u_1' and u_2' that are close to $u_{1;c}$ and $u_{2;c}$, respectively.

Under ideal conditions (no measurement errors) this attack required 570 time measurements to factorize 1024 bit moduli $n \approx 0.7 \cdot 2^{1024}$. Confirmed intervals were tried to establish after each 42 steps ([13]). When attacking a prime p_i directly (instead of any multiple) it suffices to reconstruct the upper half of the bit representation of p_1 or p_2 ([3]). For the parameters from above this reduces the number of time measurements from 570 to 300.

Also this attack may be interpreted as a sequence of decision problems with $|\Theta| = 2$, $s(1,0) = s(0,1) = 1$ and $\eta(0) = \eta(1) = 0.5$. However, the loss function and the a priori distribution do not yield any additional information in this case. We point out that this attack can be generalized to table methods ([13]) although its efficiency decreases due to a lower signal-to-noise ratio. In [1] this attack was modified to attack OpenSSL implementations over local networks.

6 A Combined Timing and Power Attack

In this section we assume that the attacked device computes modular exponentiations $y \mapsto y^d (\mathrm{mod}\, n)$ with a modular exponentiation algorithm that uses a b-bit-table ([9], Alg. 14.82) and Montgomery's multiplication algorithm (cf. Pseudoalgorithm 2). The b-bit table stores the values $\bar{y}_1, \ldots, \bar{y}_{2^b-1}$ with $\bar{y}_{j+1} = \mathrm{MM}(\bar{y}_j, \bar{y}_1; M)$ (cf. Sect. 3). We assume that the attacked device is resistant against pure power attacks but that the power measurements (SPA; cf. [16], Remark 3) enable the attacker to identify the beginning and the end of the particular Montgomery multiplications, i.e., whether an extra reduction is carried out. Due to base blinding (which prevents pure timing attacks) the attacker does not any of the table values, that is, the operands of the Montgomery multiplications. (If the attacker knew the table entries the attack was indeed elementary ([19], Subsect. 3.3).) In [19] only the special case $b = 2$ was considered. In [16] this attack was optimized and generalized to arbitrary b. Reference [17] treats the sliding windows exponentiation algorithm ([9], Alg. 14.85) with a modified variant of Montgomery's exponentiation algorithm where an extra reduction is carried out iff $s \geq R$ (cf. Sect. 3, Alg. 1). Although the general approach remains the same this increases the mathematical difficulties considerably.

The attack falls into four phases. At first the attacker measures the power consumption for a sample $y_1, \ldots, y_N$, and therefrom he determines those Montgomery operations that require extra reductions. On basis of this information he guesses the types ('S', '$M'_1, \ldots,$ 'M'_{2^b-1}) of all Montgomery operations within the exponentiation phase. The attacker guesses blocks of $f \geq 1$ consecutive Montgomery operations independently. (The attack becomes more efficient for $f > 1$ since the extra reductions of consecutive Montgomery multiplications are not independent. At the same time the computations become more complex.) Finally, the attacker tries to correct possible guessing errors and checks the resulting guess $\widetilde{d}$ for the secret exponent d (e.g. by a known digital signature).

Theorem 3 specifies the optimal decision strategy. The $\{0,1\}$-valued random variables $W_{1(k)}, W_{2(k)}, \ldots$ describe the random timing behaviour of the Montgomery multiplications in the exponentiation phase (see Sect. 3) where '(k)' indicates sample k. Equation (11) quantifies the probabilities for extra reductions which yet depend on the unknown table values. The 'source' of the attack is the initialization phase where the table values $\bar{y}_{1(k)}, \ldots, \bar{y}_{2^b-1(k)}$ computed. Although the attacker does not know the particular operands he at least knows the type of these operation ($\bar{y}_{j+1(k)} = \mathrm{MM}(\bar{y}_{j(k)}, \bar{y}_{1(k)}; M)$). The random timing behaviour in the initialization phase is quantified by another stochastic process $W'_{1(k)}, \ldots, W'_{2^b-1(k)}$ (cf. [16], Equation (3)). Theorem 3 uses Theorem 1(iii). For its proof we refer the interested reader to [16].

Theorem 3. *Let* $\tau_{opt}\Big((w_{i(k)}, \ldots, w_{i+f-1(k)}, w'_{1(k)}, \ldots, w'_{2^b-1(k)})_{1 \leq k \leq N}\Big) := \theta^*$ *if*

$$\sum_{\theta \in \Theta} s(\theta, \theta')\eta(\theta) \prod_{k=1}^{N} \mathrm{Prob}_\theta\big(W_{i(k)} = w_{i(k)}\ , \ldots, W_{i+f-1(k)} = w_{i+f-1(k)} \mid W'_{r(k)} = w'_{r(k)}, r = 1, \ldots, 2^b - 1\big)$$

is minimal for $\theta' = \theta^*$. The decision strategy τ_{opt} is optimal among all the decision strategies that guess the types $T(i), \ldots, T(i+f-1)$ simultaneously.

Apart from additional technical difficulties the conditional probabilities $\mathrm{Prob}_\theta(\cdot \mid \cdot)$ are computed in a similar manner as in (15). We refer the interested reader to Section 4 of [16]. Due to the lack of space we restrict our attention to the a priori distribution and the loss function where we exclusively consider the case $f = 1$. (The general case $f \geq 1$ is treated in [16], Sect. 5.) In particular, $\Theta = \{`S', `M'_1 \ldots, `M'_{2^b-1}\}$. In the exponentiation phase $\approx \log_2(d)$ squarings and $\approx \log_2(d)/(b2^b)$ multiplications with any particular table entry $\bar{y}_j$ are carried out. This yields the a priori distribution

$$\eta(`M'_1) = \cdots = \eta(`M'_{2^b-1}) = \frac{\frac{1}{b2^b}}{\frac{2^b-1}{b2^b}+1} = \frac{1}{b2^b(2^b-1)},\ \eta(`S') = \frac{b2^b}{b2^b(2^b-1)}. \quad (19)$$

The following example underlines that unlike in Sects. 4 and 5 it is reasonable to distinguish between different types of guessing errors.

Example 4. Let $b = 4$ and let the correct type sequence be given by
. $\ldots, `S', `M_3', `S', `S', `S', `S', `S', \quad `M_{12}', `S', `S', `S', `S', `S', `M_1', `S', \ldots$
whereas a), b) and c) are possible guesses.
a) $\ldots, `S', `M_3', `S', `S', `S', `S', `M_{11}', `M_{12}', `S', `S', `S', `S', `S', `M_1', `S', \ldots$
b) $\ldots, `S', `M_3', `S', `S', `S', `S', `S', \quad `S', \quad `S', `S', `S', `S', `S', `M_1', `S', \ldots$
c) $\ldots, `S', `M_3', `S', `S', `S', `S', `S', \quad `M_{14}', `S', `S', `S', `S', `S', `M_1', `S', \ldots$

Each of the subsequences a), b), and c) contains exactly one wrong guess. The error in Sequence a) ('M_{11}') is obvious as the number of squarings between two multiplications with table entries must be a multiple of $b = 4$. *Type-a errors* ('M_j' instead of 'S') are easy to detect and to correct if they occur isolated, i.e. if there are no further type-a or *type-b errors* ('S' instead of 'M_j'; cf. Sequence b)) in their neighbourhood. The correction of type-b-errors is not as obvious as that of type-a errors. (Reasonably, the attacker tries that alternative $a \in \Theta \setminus \{`S'\}$ that yields the second lowest expected loss.) The detection and location of type-a errors and type-b errors can be interpreted as a decoding problem. (Therefore, 'S' is replaced by 0 and 'M'_j by 1. Valid code words consist of isolated 1s and subsequences of 0s with lengths that are multiples of b.) Most cumbersome are the *type-c errors* ('M_j' instead of 'M'_t) as not even their detection is obvious. Clearly, the attacker wants to avoid false guesses. However, the optimal decision strategy need not minimize the total number of errors (which was achieved by defining $s(\theta_i, \theta_j) := 1$ for all $\theta_i \neq \theta_j \in \Theta$) but should 'favour' type-a and type-b errors in comparison with type-c errors. Consequently, it is reasonable to choose a loss function that punishes type-c errors more than type-a and type-b errors, In our practical experiments we used for $b = 4$, for instance, the values $s(`S', `M'_j) = 1$, $s(`M'_j, `S') = 1.5$, $s(`M'_t, `M'_j) = 8$ (cf. [16]). We point out that the attack can be prevented with suitable exponent blinding or constant processing times for all Montgomery operations in the exponentiation phase (cf. Remark 4 and [16], Sect. 11) but not with base blinding.

Recall that whether a Montgomery operation requires an extra reduction neither depends on the concrete hardware platform nor on the used multiprecision variant of Montgomery's multiplication algorithm but only on d, n, R and the base $y_{(k)}$ (cf. Subsect. 3.1). Hence we emulated the modular exponentiations on a computer, outputting which Montgomery operations required extra reductions. This clearly corresponds with an attack under ideal conditions (also considered in [19] and [16]) where the attacker knows definitely whether a particular Montgomery operation needs an extra reduction. We point out that the attack, though less efficient, will also work under less favourable conditions. An attack was counted as successful iff the closest code word yielded the location of all type-a and type-b errors, and if there was at most one type-c error. For RSA without CRT, $b = 2$, $n/R \approx 0.99$, $\log_2(d) \approx 384$ and $(f = 3, N = 200)$ we obtained a success rate of about 90% whereas the attack in [19] required $N = 1000$ samples. (The efficiency of the attack increases as the ratio n/R increases.) For $b = 4$, $n/R \approx 0.70$ (average case), $\log_2(d) \approx 512$ and $(f = 1, N = 550)$ about of 94% of the attacks were successful. We point out that also CRT implementations are vulnerable to this attack (cf. [16], Sect. 10).

For $b = 4$, $n/R \approx 0.70$, $\log_2(d) \approx 512$ and $(f = 1, N = 550)$, resp. $(f = 1, N = 450)$ the optimal decision strategy was successful in about 94%, resp. 67% of the trials. Neglecting the a priori distribution and the different classes of errors, i.e. when using the maximum-likelihood estimator, the success rates decreased to 74% and 12%, resp., for these two parameter sets. For the optimal decision strategy the average numbers of type-c errors per trial were about 0.3 and 0.8, respectively. When using the maximum-likelihood estimator about 0.8, resp. 2.4, type-c errors occurred per trial in average.

These results underline that the probabilities p_θ have the most significant impact on the efficiency of the decision rule. Depending on the concrete situation, however, also the a priori distribution and the definition of an appropriate loss function may have non-negligible impact on the efficiency of the decision statrategy, especially for small sample sizes.

7 A Timing Attack on a Weak AES Implementation

Reference [7] considers a timing attack on a careless AES implementation. In the MixColumn transformation multiplications over $GF(2^8)$ by '$02'$ and '$03' =$ '$01' +$ '$02'$ are carried out. Essentially, only the multiplications by '$02'$ need to be calculated, and this is done by shifting the respective state byte by one position to the left. If a carry occurs the hexadecimal constant '$1B'$ is XORed to the shifted value. In the attacked implementation these conditional operations caused differences in the encryption times since the other AES transformations required identical time for all input values. In [7] the key bytes $k_1, k_2, \ldots, k_{16}$ were treated independently, and all combinations of key byte candidates were checked by a known plaintext/ciphertext pair.

Clearly, the larger the candidate sets the more time-consuming is the checking phase. On the other hand, if a correct key byte is rejected the attack must fail.

In [15] the efficiency of this attack was increased noticeably by interpreting the encryption times as realizations of random variables and by applying statistical decision theory. The candidate sets for the particular key bytes were reduced in two steps, considering one further key byte in each step. Each reduction step itself consists of many decisions, tolerating errors in some of these individual decision problems. Due to the lack of space we omit details and refer the interested reader to [15]. We merely point out that the sample size was reduced from 48000 to 4000 with a success rate of more than 90%. Moreover, this two-step sieving process can be adjusted to other side-channel attacks (e.g., to power attacks) where different parts of the key influence the measurements simultaneously.

8 Final Remarks

This paper proposes a general method to optimize the efficiency of side-channel attacks by advanced stochastic methods, especially by applying the calculus of stochastic processes and statistical decision theory. The proposed method is not a 'ready-to-use' tool for any application but requires some work to apply it to specific problems. We yet believe that the above examples have illustrated the central principles. We emphasize that a good understanding of the potential power of an attack is necessary to be able to rate its true risk potential and to design adequate and reliable countermeasures.

References

1. D. Brumley, D. Boneh: Remote Timing Attacks are Practical. In: Proceedings of the 12th Usenix Security Symposium, 2003.
2. B. Canvel, A. Hiltgen, S. Vaudenay, M. Vuagnoux: Password Interception in a SSL/TSL Channel. In: D. Boneh (ed.): Crypto 2003, Lecture Notes in Computer Science **2729**, Springer, Heidelberg (2003), 583–599.
3. D. Coppersmith: Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. J. Cryptology **10** (no. 4) (1997) 233–260.
4. J.-F. Dhem, F. Koeune, P.-A. Leroux, P.-A. Mestré, J.-J. Quisquater, J.-L. Willems: A Practical Implementation of the Timing Attack. In: J.-J. Quisquater and B. Schneier (eds.): Smart Card – Research and Applications, Springer, Lecture Notes in Computer Science **1820**, Berlin (2000), 175–191.
5. P. Kocher: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems. In: N. Koblitz (ed.): Crypto 1996, Springer, Lecture Notes in Computer Science **1109**, Heidelberg (1996), 104–113.
6. K. Gandolfi, C. Mourtel, F. Olivier: Electromagnetic Analysis: Concrete Results. In: Ç.K. Koç, D. Naccache, C. Paar (eds.): Cryptographic Hardware and Embedded Systems – CHES 2001, Springer, Lecture Notes in Computer Science **2162**, Berlin (2001), 251–261.
7. F. Koeune, J.-J. Quisquater: A Timing Attack against Rijndael. Catholic University of Louvain, Crypto Group, Technical report CG-1999/1, 1999.
8. P. Kocher, J. Jaffe, B. Jub: Differential Power Analysis. In: M. Wiener (ed.): Crypto 1999, Springer, Lecture Notes in Computer Science **1666**, Berlin (1999), 388–397.

9. A.J. Menezes, P.C. van Oorschot, S.C. Vanstone: Handbook of Applied Cryptography, Boca Raton, CRC Press (1997).
10. D. Neuenschwander: Probabilistic and Statistical Methods in Cryptology. An Introduction by Selected Topics. Springer, Lecture Notes in Computer Science **3028**, Berlin (2004).
11. H. Sato, D. Schepers, T. Takagi: Exact Analysis of Montgomery Multiplication. TU Darmstadt, Technical Report TI-6/04.
12. W. Schindler: Optimized Timing Attacks against Public Key Cryptosystems. Statist. Decisions **20** (2002), 191–210.
13. W. Schindler: A Timing Attack against RSA with the Chinese Remainder Theorem. In: Ç.K. Koç, C. Paar (eds.): Cryptographic Hardware and Embedded Systems – CHES 2000, Springer, Lecture Notes in Computer Science **1965**, Berlin (2000), 110–125.
14. W. Schindler, F. Koeune, J.-J. Quisquater: Unleashing the Full Power of Timing Attack. Catholic University of Louvain, Technical Report CG-2001/3.
15. W. Schindler, F. Koeune, J.-J. Quisquater: Improving Divide and Conquer Attacks Against Cryptosystems by Better Error Detection / Correction Strategies. In: B. Honary (ed.): Cryptography and Coding – IMA 2001, Springer, Lecture Notes in Computer Science **2260**, Berlin (2001), 245–267.
16. W. Schindler: A Combined Timing and Power Attack. In: P. Paillier, D. Naccache (eds.): Public Key Cryptography – PKC 2002, Springer, Lecture Notes in Computer Science **2274**, Berlin (2002), 263–279.
17. W. Schindler, C. Walter: More Detail for a Combined Timing and Power Attack against Implementations of RSA. In: K.G. Paterson (ed.): Cryptography and Coding – IMA 2003, Springer, Lecture Notes in Computer Science **2898**, Berlin (2003), 245–263.
18. C.D. Walter: Precise Bounds for Montgomery Montgomery Modular Multiplication and Some Potentially Insecure RSA Moduli. In: B. Preneel (ed.): Topics in Cryptology – CT-RSA 2002, Springer, Lecture Notes in Computer Science **2271**, Berlin (2002), 30–39.
19. C.D. Walter, S. Thompson: Distinguishing Exponent Digits by Observing Modular Subtractions. In: D. Naccache (ed.): Topics in Cryptology – CT-RSA 2001, Springer, Lecture Notes in Computer Science **2020**, Berlin (2001), 192–207.
20. H. Witting.: Mathematische Statistik I, Stuttgart, Teubner (1985).