Revisiting PACD-based Attacks on RSA-CRT

Guillaume Barbu[1], Laurent Grémy[1] and Roch Lescuyer[2]

[1] IDEMIA, Pessac, France
[2] IDEMIA, Courbevoie, France

Abstract. In this work, we use some recent developments in lattice-based cryptanalytic tools to revisit a fault attack on RSA-CRT signatures based on the Partial Approximate Common Divisor (PACD) problem. By reducing the PACD to a Hidden Number Problem (HNP) instance, we decrease the number of required faulted bits from 32 to 7 in the case of a 1024-bit RSA. We successfully apply the attack to RSA instances up to 8192-bit and present an enhanced analysis of the error-tolerance in the Bounded Distance Decoding (BDD) with predicate approach. Finally, evaluating the impact of standard side-channel and fault countermeasures, we show that merely verifying the signature before output is not an adequate protection against this attack. The reduction from PACD to HNP might be of independent interest.

Keywords: Lattice reductions · RSA-CRT · PACD · HNP · BDD with Predicate

1 Introduction

Lattice reduction algorithms aim at transforming a given lattice basis into a reduced form, typically resulting in shorter and more orthogonal basis vectors. By doing so, lattice reductions facilitate the resolution of difficult problems, such as finding the shortest vector in a lattice (Shortest Vector Problem, SVP) or a vector in the lattice particularly close to another vector (Closest Vector Problem, CVP). Notably, lattice reduction algorithms like LLL [LLL82] and BKZ [SE94] have found widespread applications due to their efficiency and effectiveness in reducing lattice bases [Sch87, SE94, GN08, MW16]. The field of lattice reduction is still very active. For instance, recently, Ryan and Heninger proposed a novel algorithm [RH23] that allows to reduce lattices of a very large dimension which were previously considered out of reach, along with a public implementation called `flatter` [Rya23].

A well-known application of lattices as a cryptanalytic tool is to break the ECDSA signature scheme, through a reduction to the Hidden Number Problem (HNP). The latter is to determine an integer α given many samples (a_i, t_i) where $a_i + k_i = \alpha \cdot t_i \bmod N$ for which we know that $k_i < 2^l$ for some bound l, N being a public modulus. Considering an ad-hoc lattice, an HNP instance can be viewed as a CVP instance. It allows to break ECDSA signatures when a few bits of the nonce are known. Indeed, an ECDSA signature (r, s) is computed as $s = k^{-1} \cdot (h + r \cdot d) \bmod N$ where r is the x-coordinate of $[k]G$, k is the nonce and h a (hash of a) message. If we know the most significant bits u of the nonce $k = u \cdot 2^l + v$, where $v < 2^l$, for an ECDSA signature (r, s) on a message h, we have that:

$$(-s^{-1} \cdot h + u \cdot 2^l) + v = d \cdot (s^{-1} \cdot r) \bmod N,$$

which is an HNP instance for the secret key d. Depending on the number of signatures and the number of known bits, such an instance might be tractable.

E-mail: guillaume.barbu@idemia.com (Guillaume Barbu), laurent.gremy@idemia.com (Laurent Grémy), roch.lescuyer@idemia.com (Roch Lescuyer)

Bounded Distance Decoding (BDD) is a variant of CVP. It consists in finding a vector close to another vector with the additional guarantee that the target vector is not too far from a lattice element. Recently, the authors of [AH21] note that in some applications there is a mean to check whether the valid solution is one of the vectors returned by the lattice reduction. Hence the idea of adding a *predicate* check inside the lattice algorithm. This is called the BDD with predicate approach. In the case of the ECDSA scheme, one can check that a value α is a valid solution, since the public key (G, P) is available. The predicate in this case is basically a check that the public key is compatible with the value α, namely a check that $P \stackrel{?}{=} [\alpha]G$.

In this paper, we turn our attention to RSA-CRT signatures. Given a public modulus N, a private key d and a message m, it is well-known that the generation of an RSA signature $s = m^d \bmod N$ can also be computed by the so-called CRT (Chinese Remainder Theorem) trick [QC82] as

$$s = s_q + q \cdot [(s_p - s_q) \cdot i_q \bmod p],$$

where s_p and s_q are the results of the modular exponentiations of the message modulo p and q, the two secret factors of the RSA modulus N, respectively, and i_q the inverse of q modulo p.

Such signatures are particularly sensitive to fault attacks, which represent an important threat to the security of cryptographic hardware and embedded software. From their introduction in [Bel96], where the authors showed that the slightest error during an RSA signature generation can allow to compute the factorization of N with a simple *greatest common divisor* (GCD) computation, fault attacks have found countless applications, breaking all kinds of cryptosystems [BS97, PQ03, DLV03, CJ05, LFZD16]. The literature on fault attacks and countermeasures concerning RSA implementations is very important [Len96, JQBD97, BDL01, ABF+03, Gir06, YKM06, KQ07, BCG08, Vig08, CJK+09, EL10, RG14].

At CHES 2012, Fouque et al. proposed [FGL+12] a novel exploitation of faulty RSA signatures [RSA78]. The authors see the RSA-CRT recombination as a Partial Approximate Common Divisor (Partial ACD, or PACD). Thanks to a fault injection during one of the modular exponentiations of the RSA-CRT setting 32 bits of the result to 0, they put themselves in a position where they can solve this PACD instance and thus find the secret factors p and q.

Contributions. In this work, we revisit the fault attack of [FGL+12], leveraging the BDD with predicate approach and `flatter`, and show that the knowledge of only 7 bits of the partial result is sufficient to factorize a 1024-bit RSA modulus instead of the 32 bits required previously. Our approach benefits here from a reduction from the PACD problem to the HNP, which is not a general result, but holds for our particular instance. Furthermore, we show that the attack remains practical on larger RSA instances with experimental tests up to 8192-bit RSA.

Another aspect of the BDD with predicate approach compared to the state of the art is the ability to withstand some errors in the bits that are supposed to be known. Until this breakthrough, the lattice approach to solving BDD-like problems was considered useless when such errors were likely. However, in their article [AH21], Albrecht and Heninger only consider the case where one bit was erroneous, i.e. when the recovered length of the nonce was wrong by 1. In this work, we consider errors that may affect each one of the recovered bits, and explore how the BDD with predicate approach can be adapted in this case.

Finally, from our refined results, we conclude that the verification of the signature before releasing it does *not* provide a satisfactory protection. We then evaluate the robustness of other popular countermeasures in the field of embedded cryptography, exhibiting some interesting solutions based on randomized blinding.

Organisation. After some preliminaries (Section 2), we describe our main procedure to attack RSA-CRT signatures (Section 3), analyse the effect of errors on the known bits (Section 4). Finally, we discuss the effect of some standard counter-measures against our attack (Section 5) before we conclude (Section 6).

2 Preliminaries

One of the primary tools to solve both the PACD and HNP problems relies on lattices, which are discrete subgroups of $\mathbb{R}^n$. The elements of a full rank-n lattice Λ are the integer linear combinations of $\mathcal{B} = \{\boldsymbol{b}_0, \boldsymbol{b}_1, \dots, \boldsymbol{b}_{n-1}\}$, where the $\boldsymbol{b}_i$ are linearly independent. The set $\mathcal{B}$ is called a basis of Λ, often represented by a matrix $\mathbf{B}$ for which the rows are the $\boldsymbol{b}_i$. We denote by the application $\mathcal{L}$ the fact that the lattice Λ is generated by the rows of $\mathbf{B}$, i.e. $\Lambda = \mathcal{L}(\mathbf{B})$. There exists infinitely many bases of Λ, but a convenient way to solve hard lattice problems, e.g. shortest vector problem (SVP) or bounded distance decoding (BDD), is to work with reduced bases, informally bases with short orthogonal vectors, which can be obtained using lattice basis reduction algorithms [LLL82, SE94, NS09, CN11, RH23].

The volume of a lattice Λ is denoted by $\det(\Lambda)$ and is equal to $\det(\mathbf{B})$, which is an invariant of the lattice. For a *random lattice* Λ [Ajt96], the Gaussian heuristic $\mathrm{gh}(\Lambda)$ gives a bound for the Euclidean length λ_1 of the shortest non-zero vector

$$\lambda_1 < \mathrm{gh}(\Lambda) = \sqrt{n/(2\pi e)}\,\det(\Lambda)^{1/n}.$$

A way to get (an approximation of) the shortest vector, and then solve the (approximate) SVP, is to look at the first basis vector returned by a lattice basis reduction algorithm. A (δ, η)-LLL reduced lattice basis [NS09], where $1/4 < \delta < 1$ and $1/2 < \eta < \sqrt{\delta}$, is composed of a vector $\boldsymbol{b}_0$ for which its Euclidean norm is bounded by

$$\|\boldsymbol{b}_0\| \leq (1/(\delta - \eta^2))^{(n-1)/4}\,\det(\Lambda)^{1/n} = R^n\,\det(\Lambda)^{1/n}, \tag{1}$$

where R is the root Hermite factor. The usual choice $\delta = 0.99$ and $\eta = 0.51$ gives $R \approx 1.075$, which is larger than what is practically expected in the average case where $R \approx 1.02$ [NS06]. An α-lattice-reduced basis as defined in [RH23] reaches practically the same bound, where $\alpha \approx 2\log_2(R)$ [Rya23]. The difference between the two algorithms lies in the running time: for a lattice basis with entries of size p, the running time of [NS09] is $O(n^{4+\epsilon}(p + \log n)(p + n))$ compared to $O(n^\omega(p + n)^{1+\epsilon})$ for [RH23], where $\omega \in [2, 3)$ is the matrix multiplication exponent.

These two polynomial-time algorithms may return for large dimension lattices a short vector not sufficiently accurate compared to the Gaussian heuristic prediction, which may fail to help solving the PACD or HNP problems. It is then needed to use exponential time algorithms. Two are of main interest: the enumeration algorithm, which find a short vector of norm less than $\gamma \cdot \mathrm{gh}(\Lambda)$ in $\gamma^n n^{n/(2e)+o(d)}$ steps [HS07], and the sieving algorithm which find $||\boldsymbol{b}_0|| \leq \sqrt{4/3}\,\mathrm{gh}(\Lambda)$ in $2^{0.292+o(1)}$ steps [BDGL16, ADH+19].

Both these algorithms may be used as SVP oracles inside BKZ [Sch87, SE94, CN11] on blocks of dimension k. When BKZ is instantiated with the enumeration algorithm and n is sufficiently large compared to k, BKZ outputs a short vector of norm bounded by Equation 1 where $R \approx k^{1/(2k)}$ in time $k^{k/(2e)+o(k)}$ [HS07]. In our context, you may refer to [AH21] for technical details. Practically, we can find the use of BKZ with an enumeration algorithm in [dt23] and BKZ with a sieving algorithm in the companion implementation of [AH21].

3 Attacking RSA-CRT Signatures through PACD

In this section, we recall the definition of the ACD problem (Section 3.1), the principle of the PACD-based fault attack from [FGL+12] and define our attacker setting (Section 3.2), describe the state-of-the-art approach to solve the PACD (Section 3.3), show how to fall back to a HNP instance (Section 3.4), and finally report on some experiments (Section 3.5).

3.1 The Approximate Common Divisor Problem

An ACD instance is given by a set of samples $\{s_i\}_i$ where

$$s_i = r_i + \alpha \cdot p_i \tag{2}$$

for a secret integer α. Such an instance is parameterized by the sizes of the target α, and of the r_i and the p_i components. In the Partial variant of the problem, an exact multiple of the target α is also given (in other words, there exists a sample such that $r_i = 0$). The ACD problem has been thoroughly analysed in [GGM16] where several resolution methods are compared.

3.2 RSA-CRT as a Partial ACD instance

RSA-CRT Samples. Let us recall the recombination of the partial RSA-CRT signatures. An RSA signature $s = m^d \bmod N$ can be computed as:

$$d_q = d \bmod (q-1), \qquad d_p = d \bmod (p-1), \qquad i_q = q^{-1} \bmod p,$$

$$s_q = m^{d_q} \bmod q, \qquad s_p = m^{d_p} \bmod p, \qquad s = s_q + q \cdot [(s_p - s_q) \cdot i_q \bmod p].$$

Gathering several signatures s_i for different m_i can be seen as an instance of the Partial ACD problem using the notation introduced in Equation 2

$$s_i = \underbrace{s_q}_{r_i} + q \cdot \underbrace{[(s_p - s_q) \cdot i_q \bmod p]}_{p_i},$$

where the prime factor q is the target integer.

Attacker Setting. If an attacker were capable of knowing – or forcing – the most significant bits of s_q to a known value, the problem would become tractable.

Let $\widetilde{s}$ be a signature for which we know that the Most Significant Bits (MSB) of $\widetilde{s_q}$ are known. For simplicity, let us consider that they are null. For instance, we can consider the case of a fault injection that sets the most significant bits of s_q to zero, leading to a faulty $\widetilde{s_q}$. We get the following relation for the faulty signature $\widetilde{s}$:

$$\widetilde{s} = \widetilde{s_q} + q \cdot [(s_p - s_q) \cdot i_q \bmod p].$$

We then know that $\widetilde{s_q} < \frac{q}{2^l}$ for some l. With a sufficient number of signatures, it becomes possible to solve the Partial ACD and thus recover the target value q and thus the factorisation of the RSA modulus N.

We note that a direct exploitation of such a fault is possible if $\widetilde{s_q}$ is completely zeroized. In this case, computing the greatest common divisor of N and $\widetilde{s}$ reveals the prime factor q.

In the rest of this section, we bound p and q such that they are η-bit primes, then the p_i are at most η-bit integers. We also bound r_i to be a ρ-bit integer, that is $\rho = \eta - l$.

3.3 The Simultaneous Diophantine Approximation Approach

Several methods are described in the state-of-the-art to solve ACD instances [GGM16]. We can cite the Simultaneous Diophantine Approximation (SDA) approach [How01], the Orthogonal based (OL) approach [NS01], and the Multivariate polynomial (MP) approach [CH13].

Why SDA. In our case, we choose to pick the SDA approach, for the following reasons. On the one hand, according to [GGM16], the MP approach does not outperform the other approaches. On the other hand, the condition for success of the OL approach given in [GGM16, Section 4] is not met in our particular case for reasonable RSA-CRT parameters and number of known bits.

SDA attack idea. According to the SDA approach, we let $\mathcal{L}(\mathbf{M})$ be the lattice in $\mathbb{Z}^{t+1}$ spawn by

$$\mathbf{M} = \begin{pmatrix} 2^\rho & s_1 & \dots & s_t \\ & -s_0 & & \\ & & \ddots & \\ & & & -s_0 \end{pmatrix}.$$

We note that $\mathcal{L}(\mathbf{M})$ contains the vector

$$\begin{aligned} \boldsymbol{v} &= (p_0, p_1, p_2, \dots, p_t) \cdot \mathbf{M}, \\ &= (2^\rho \cdot p_0, p_0 \cdot s_1 - s_0 \cdot p_1, p_0 \cdot s_2 - s_0 \cdot p_2, \dots, p_0 \cdot s_t - s_0 \cdot p_t), \\ &= (2^\rho \cdot p_0, p_0 \cdot r_1 - r_0 \cdot p_1, p_0 \cdot r_2 - r_0 \cdot p_2, \dots, p_0 \cdot r_t - r_0 \cdot p_t), \end{aligned}$$

since $p_0 \cdot s_i - p_i \cdot s_0 = p_0 \cdot (q \cdot p_i + r_i) - p_i(q \cdot p_0 + r_0) = p_0 \cdot r_i - p_i \cdot r_0$. The Gaussian heuristic let us expect that the shortest vector in $\mathcal{L}(\mathbf{M})$ has norm close to

$$\sqrt{\frac{t+1}{2\pi e}} \cdot 2^\rho \cdot ({s_0}^t)^{1/(t+1)} \approx \sqrt{t+1} \cdot 2^{(\rho + 2 \cdot t \cdot \eta)/(t+1)}.$$

Depending on ρ, we can then expect that $\boldsymbol{v}$ is revealed by reducing $\mathcal{L}(\mathbf{M})$. This will give a candidate value for p_0 which will in turn allows to recover the target value $q = (s_0 - (s_0 \bmod p_0))/p_0$.

Improvement. We further note that to maximize our chances of success, it is desirable to select the largest value s_i as s_0 (the one on the diagonal of $\mathcal{L}(\mathbf{M})$). Indeed, the larger s_0, the larger the determinant of $\mathcal{L}(\mathbf{M})$, and hence the larger the Gaussian heuristic bound.

In our RSA-CRT context, we know that all the $s_i < N$, and we know that N is an exact multiple of q. This lets us update our lattice basis $\mathbf{M}$ as follows

$$\mathbf{M} = \begin{pmatrix} 2^\rho & s_1 & \dots & s_t \\ & -N & & \\ & & \ddots & \\ & & & -N \end{pmatrix},$$

and our expected small vector is then $\boldsymbol{v} = (2^\rho \cdot q, q \cdot r_1, q \cdot r_2, \dots, q \cdot r_t)$, since $N = q \cdot p_0 + r_0$, where $p_0 = p$ and $r_0 = 0$.

3.4 The Hidden Number Problem Is Back

In this sections, we show how our PACD instance can heuristically be viewed as an HNP instance, and give a more formal analysis of this reduction.

3.4.1 Using the HNP in our Attack Setting

Interestingly, the last basis is similar to a basis generally used to solve HNP instances, up to a different labelling of the lines and columns of the matrix. Indeed we can also see our problem as an HNP instance

$$s_i \cdot p \bmod N < \frac{N}{2^l},$$

where the $(s_i)_i$ are known, and p is the target hidden number.

According to the Gaussian heuristic, the shortest vector in the lattice is expected to have norm

$$\mathrm{gh}(\mathcal{L}(\mathbf{M})) \leq \sqrt{\frac{t+1}{2\pi e}}(N^t \cdot 2^\rho)^{1/(t+1)}.$$

We can then study the expectation for the norm of the target vector $\boldsymbol{v}$, to assess our chance of success depending on the number of known bits. In [GGM16], the authors have already proved that in the generic ACD setting, $\|\boldsymbol{v}\|$ has a tight upper bound of

$$0.47 \cdot \frac{\sqrt{t+1}}{q} \cdot 2^{\rho-1+\gamma}.$$

In our setting, the main differences are that $p_0 = p$, $r_0 = 0$ and the r_i are taken from $[0, 2^\rho)$, whereas in [GGM16] they are taken from $(-2^\rho, 2^\rho)$. We then have

$$\begin{aligned}\mathbb{E}(\|\boldsymbol{v}\|^2) &= \mathbb{E}\left(\sum_{i=1}^{t}(q \cdot r_i)^2 + (q \cdot 2^{\rho+1})^2\right), \\ &= t \cdot q^2 \cdot \mathbb{E}({r_i}^2) + q^2 \cdot 2^{2\cdot\rho+2},\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}({r_i}^2) &= \frac{1}{2^{\rho+1}} \cdot \sum_{i=0}^{2^{\rho+1}} i^2, \\ &= \frac{1}{2^{\rho+1}} \cdot \left(\frac{1}{3} \cdot 2^{3\rho+3} + \frac{1}{3} \cdot 2^\rho + 2^{2\cdot\rho+1}\right), \\ &= \frac{1}{3} \cdot 2^{2\cdot\rho+2} + \frac{1}{6} + 2^\rho, \\ &\approx \frac{1}{3} \cdot 2^{2\cdot\rho+2}.\end{aligned}$$

As a result,

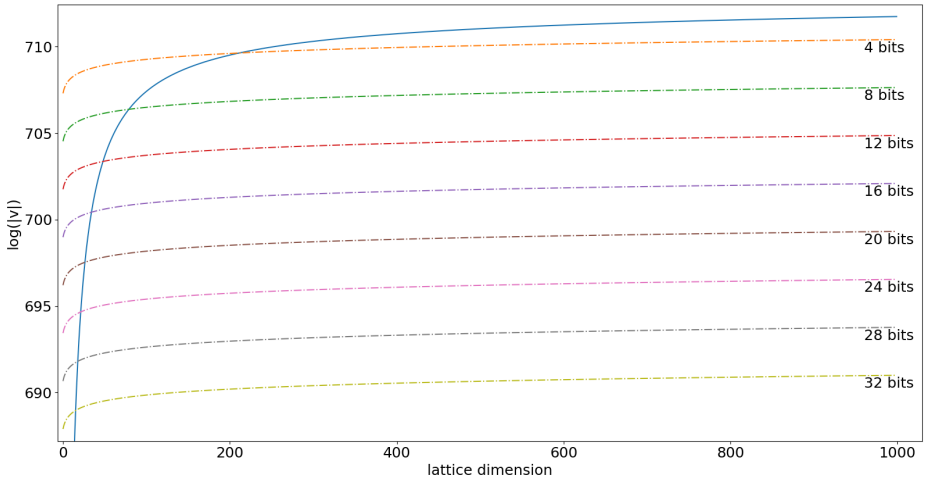
$$\mathbb{E}(\|\boldsymbol{v}\|^2) \approx t \cdot q^2 \cdot \frac{1}{3} \cdot 2^{2\cdot\rho+2} + q^2 \cdot 2^{2\cdot\rho+2} \approx \frac{1}{3} \cdot (t+1) \cdot q^2 \cdot 2^{2\cdot\rho+2},$$

and Jensen's inequality shows that $\mathbb{E}(\|\boldsymbol{v}\|) \leq \sqrt{\mathbb{E}(\|\boldsymbol{v}\|^2)} \approx 0.58 \cdot \sqrt{t+1} \cdot q \cdot 2^{\rho+1}$.

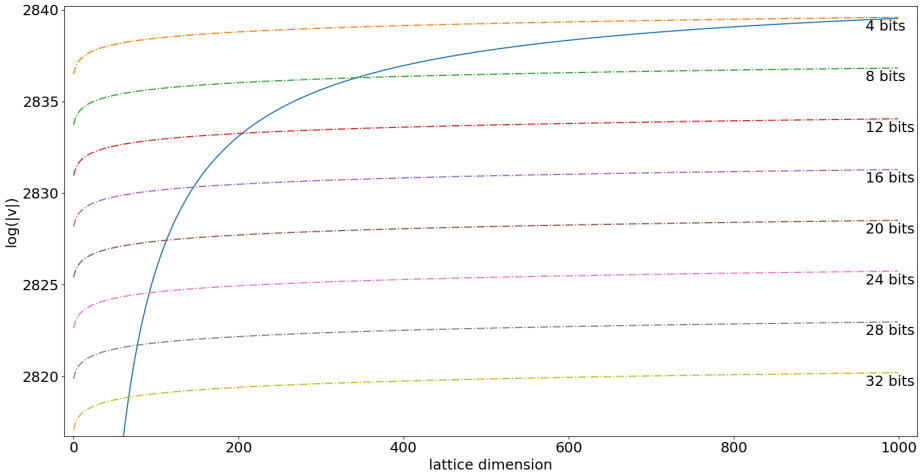
We can then confront the Gaussian heuristic with the expected norm of our target vector for different lattice dimensions and number of known bits. This is illustrated in Figure 1 for the case of an RSA-1024 modulus ($\log_2(p) = 512$), and an RSA-4096 modulus ($\log_2(p) = 2048$). According to the curves, we can then expect to have a successful attack with

- 16, 22, 35, 79 and 214 signatures for 32 bits, 24 bits, 16 bits, 8 bits and 4 bits in the case of RSA-1024,

- 68, 94, 146, and 344 signatures for 32 bits, 24 bits, 16 bits and 8 bits in the case of RSA-4096.



(a) RSA-1024.



(b) RSA-4096.

Figure 1: Evolution of the Gaussian heuristic for various lattice dimensions (plain), compared to the evolution of the expected norm for the target vector (dashed) for different number of known bits (from 4 to 32).

3.4.2 From a Reductionist Point of View

In this subsection, we have a closer look at the reduction we used in the previous subsection to solve the Partial ACD problem.

Let us begin with some notations. Let n be a modulus. Let $x \in \mathbb{Z}$ be an integer. We denote by $\lfloor x \rceil_n$ the value x modulo n in the range $[0, n)$ and by $|x|_n$ the value x modulo n in the range $\mathbb{Z} \cap [-\frac{n}{2}, \frac{n}{2})$. The set $\mathrm{MSB}_{\ell,n}(x)$ is the set of integers u satisfying $|x - u|_n < n/2^{\ell+1}$.

Let $\gamma, \eta, \rho \in \mathbb{N}$ be three integers. Let β be an η-bit odd integer. We have: $2^{\eta-1} < \beta < 2^{\eta}$. Let $\mathcal{D}$ be the following distribution, parametrized by γ, ρ.

$$\mathcal{D}_{\gamma,\rho}(\beta) := \{\beta \cdot m + r \mid m \leftarrow \mathbb{Z} \cap [0, 2^\gamma/\beta), r \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}\,.$$

Now we can formally state the ACD and Partial ACD problems.

Problem 1 (ACD)**.** *Given polynomially many samples x_i from $\mathcal{D}_{\gamma,\rho}(\beta)$, compute β.*

Problem 2 (PACD)**.** *Given polynomially many samples x_i from $\mathcal{D}_{\gamma,\rho}(\beta)$, and a sample $x_0 = \beta \cdot m_0$ for a uniformly chosen $m_0 \in \mathbb{Z} \cap [0, 2^\gamma/\beta)$, compute β.*

Let us have a look at the HNP problem. Let α be a value in $\mathbb{Z}_n$, for a public modulus n. Let $\ell \in \mathbb{N}$ be an integer. Let $\mathcal{H}$ be the following distribution, parametrized by n and ℓ.

$$\mathcal{H}_{n,\ell}(\alpha) := \{(t_i, u_i) \mid t_i \leftarrow [1, n), s_i = \lfloor t_i \cdot \alpha \rfloor_n, u_i \in \mathrm{MSB}_{\ell,n}(s_i)\}\,.$$

Problem 3 (HNP)**.** *Given polynomially many samples (t_i, u_i) from $\mathcal{H}_{n,\ell}(\alpha)$, compute α.*

The reduction from PACD to HNP we have in mind is the following one. Let $\mathcal{B}$ be an algorithm that solves the (n, ℓ)-HNP problem in time t with probability ϵ. We want to construct an algorithm $\mathcal{A}$ that solves the (γ, η, ρ)-PACD problem in time t' with probability ϵ', for some quantities to be defined.

Let $X := \{x_i\}$ be a set of samples for the Partial ACD problem. Let $n = x_0$. Construct a set of HNP samples $\{(t_i, u_i)\}$ as $t_i = x_i$, $u_i = 0$. Give the set of samples to $\mathcal{B}$, which eventually returns a value α. If α divides x_0, then algorithm $\mathcal{A}$ sets $\beta = x_0/\alpha$, otherwise $\mathcal{A}$ returns $\perp$.

For some constraints on the parameter this reduction is correct. Indeed, we know that $x_0 = \beta \cdot m_0$. If we multiply each sample x_i by m_0, we get that $x_i \cdot m_0 = \beta \cdot m_i \cdot m_0 + r_i \cdot m_0$. If we have that $|r_i| < 2^\rho < 2^{\eta-1}$, then we have that $\lfloor x_i \cdot m_0 \rfloor_n = r_i \cdot m_0$. We can bound this quantity by $r_i \cdot m_0 < 2^{\gamma+\rho}/\beta < 2^{\gamma+\rho-\eta+1}$. Hence, by setting $u_i = 0$, we have that $u_i \in \mathrm{MSB}_{\ell,n}(m_0)$ for $n = x_0$ and $\ell < \delta + \eta - \gamma - \rho - 3$ where δ is the size in bits of x_0. Indeed:

$$\lfloor x_i \cdot m_0 \rfloor_n < 2^{\gamma+\rho-\eta+1} < 2^{\delta-1-\ell-1} < x_0/2^{\ell+1}.$$

Therefore, the $\{(t_i, u_i)\}$ are samples targetting the unknown $\alpha = m_0$. Once α is found, the target β equals x_0/α.

The problem here is that we cannot argue on the probability of success ϵ' of $\mathcal{A}$. Let us explain why. The HNP problem asks the t_i to be uniformly random. However, our t_i are clearly not uniformly random. Moreover, their structure is what makes them possibly solvable, and it is hard to randomize them before giving them to $\mathcal{B}$ without breaking this structure. So we cannot claim a reduction between those problems, in all their generality.

However, in our case we have some additional assumptions. In particular, the partial ACD modulus x_0 is greater than all the other x_i samples, which are partial RSA signatures. And, even if it is hard to argue about the distribution of the signatures, they are sufficiently random in $\mathbb{Z}_{x_0}$ so that our reduction heuristically works in our case.

3.5 Experiments

Experiments with `flatter`. The experiments with `flatter` [RH23] were very straightforward, we just use `flatter` to perform the lattice reduction. Success rate are computed for 100 iterations. We obtained successful attacks for

- 7 bits known for RSA 1024, 63% success rate with 2500 signatures,
- 10 bits known for RSA 2048, 96% success rate with 1500 signatures,
- 13 bits known for RSA 4096, 11% success rate with 2500 signatures,

- 20 bits known for RSA 8192, 70% success rate with 2500 signatures.

The longest execution runs approximatively 70h for 100 lattice reductions of a lattice of dimension 2500, and 8192-bit elements (64 cores). Figure 2 shows the evolution of the success rate with respect to the lattice dimension for RSA 1024.

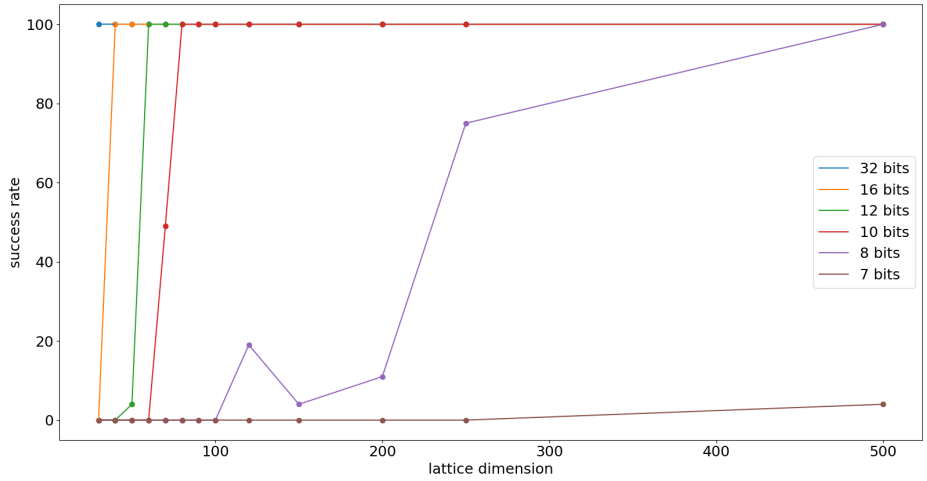


Figure 2: Evolution of the success rate for 100 iterations when attacking RSA-1024 with a standard lattice reduction using `flatter`.

Experiments with BDD with Predicate. The experiments with the code available for BDD with Predicate [AH21] required some modifications. First, we adapted the code to work for RSA-CRT. We implemented a new *Solver* with a specific predicate to check. The predicate in our case is a GCD computation to check that the modulus N can be factorized by the solution. We also had troubleshoot precision issues because we work with large integers.

Figure 3 show the evolution of the success rate with respect to the lattice dimension for RSA 1024.

Comparison between `flatter` and BDD with Predicate. Figure 4 shows the difference between our experiments with `flatter` as reduction, and our experiments with the BDD with Predicate approach. When less bits are available, we see a clear gain when we add a predicate into the lattice reduction. This is understandable, given that the algorithms used in the BDD with Predicate code ensures a better reduction quality than `flatter`'s. On the other hand, the capacity of `flatter` to handle lattices of larger dimension allows to reach our lower bound in terms of required number of known bits. Indeed, the attack succeeds with 7 known bits, while 32 known bits were used in the attack of [FGL+12].

Handling more cases. For simplicity, we focused on the case where the MSB of the s_q partial signature were null. Standard techniques enables to extend to the Known MSB / LSB cases. For instance, the BDD with Predicate repository has been enhanced to handle such cases [She21]. The idea is simply to cast it back to the case MSB at 0 by (i) subtracting the known value in the MSB case, and (ii) subtracting the known value and multiply by the inverse of 2^w in the LSB case.

In addition, we note that other variables can be targeted by our attack. Obviously, our analysis still applies when considering the other partial signature s_p. But one can also

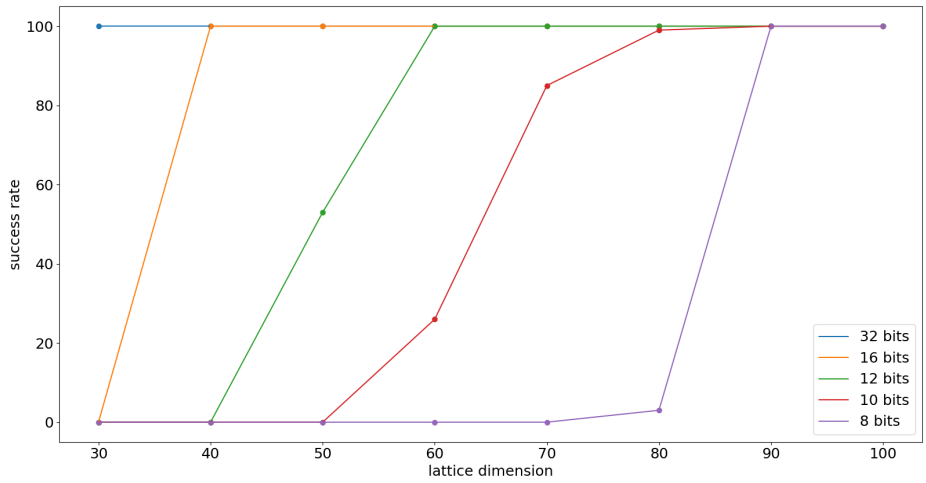


Figure 3: Evolution of the success rate for 100 iterations when attacking RSA-1024 with the BDD with Predicate approach.

Table 1: Expected number of iterations for different prediction error rate.

Error rate	0.001	0.005	0.01	0.02	0.05	0.1
Iterations	18	1140	346104	$\approx 10^9$	$\approx 10^{18}$	$\approx 10^{31}$

consider to fault m_q (or m_p), the reduction modulo q (or modulo p) of the message m before the exponentiation since one can also use the relation

$$\widetilde{s^e} = \widetilde{m_q} + q \cdot [(m_p - \widetilde{m_q}) \cdot i_q \bmod p].$$

4 Evaluating the Impact of Errors

In the lattice-reduction based attack, the widespread approach when some errors exist in the predictions of MSB/LSB consists in considering a sufficiently large set of signatures and repeat the attack on subsamples taken from this input pool. The intuitive idea is that, depending on the prediction error rate, the attacker will eventually peek a subsample with only correct predictions, and thus the attack will succeed.

For an attack based on an l-bit bias, and a given prediction error rate p_ε, one should then consider a minimal set of $t' > t/(1-p_\varepsilon)^l$ signatures, with t the number of samples required for the attack to work when the predictions are all correct. The probability to sample t signatures with correct predictions out of the pool of t' signatures follows a hypergeometric distribution and is then:

$$P[t \text{ error-free samples}] = \frac{\binom{\lfloor t\cdot(1-p_\varepsilon)^l \rfloor}{t}}{\binom{t'}{t}}.$$

Consequently, we can give an estimation for the number of iterations needed before one can expect to sample a prediction-error-free subsample and thus recover the secret value thanks to the lattice reduction: $\frac{\binom{t'}{t}}{\lfloor\binom{t\cdot(1-p_\varepsilon)^l \rfloor}{t}}$. This is illustrated in Table 1 for the attack reported in [FGL+12]: $l = 32$ and $t = 17$.

We can see from Table 1 that this approach rapidly becomes intractable when the error rate rises. With an average runtime of 3 seconds for executing one such lattice reduction

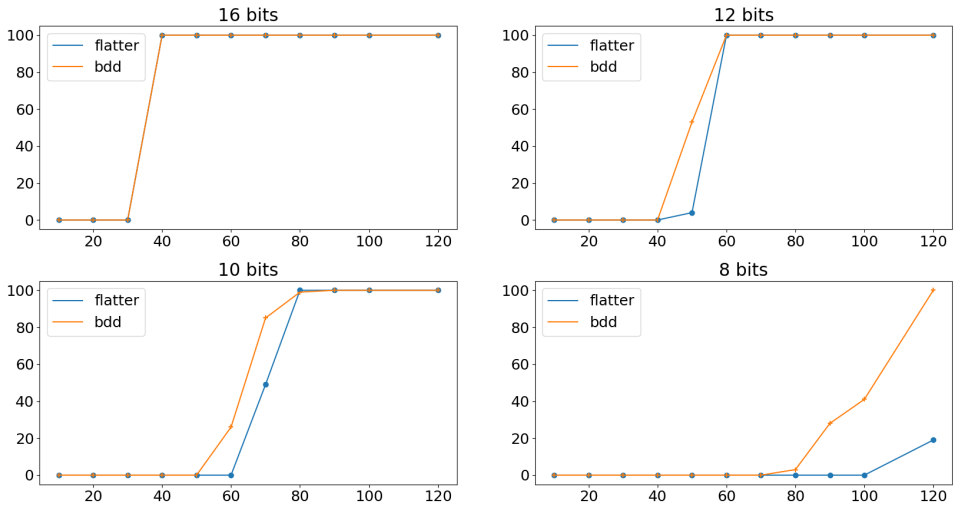


Figure 4: Comparison between applying a standard reduction with `flatter` and the BDD with Predicate approaches.

on our machine, it would take us more than 12 days to have a successful attack when the error rate is 0.01 ; and more than 110 years with an error rate of 0.02.

In [AH21], the authors assess the resilience of their approach to errors by measuring the success rate of the attack when the scalar is actually one bit longer than predicted. This is relevant when the side-channel used is likely to reveal the bit-length of the nonce, i.e. when it reveals that the MSB is 0. When the leakage rather reveals the value of the MSB or of the LSB of that nonce, with some independent errors on each bit, for instance when a fault injection has a certain probability of achieving the desired effect, another analysis is required. The intuition is that if the errors impact some of the low bits of the MSB part, it will have a small impact on the radius increase of the hyperball in which we are searching for a solution. On the other hand, if we produce erroneous predictions in the most significant bits, then the solution will not be recovered efficiently.

An important factor in the strategy of the BDD with predicate is the expected norm of the target vector. Let X be the random variable representing the value of the r_i^2, as denoted in Section 3.4. We can express the expectation for X as

$$\mathbb{E}(X) = \sum_{y=0}^{l} P(Y = y)\mathbb{E}(X|Y = y)\,,$$

where Y is the random variable denoting the number of errors in the MSB of X, for which we have

$$P(Y = y) = \binom{l}{y} p_\varepsilon^y (1 - p_\varepsilon)^{l-y}\,.$$

We note that the value of a coordinate of the target vector with y erroneous predictions of 0-MSB is $r_i + z2^\rho$, where z of Hamming Weight $y = \mathrm{HW}(z)$. It follows that

$$\mathbb{E}(X|Y = y) = \frac{1}{\binom{l}{y}} \left(\sum_{\substack{z=0 \\ \mathrm{HW}(z)=y}}^{2^l-1} \frac{1}{2^\rho} \sum_{i=0}^{2^\rho-1} (i + z2^\rho)^2 \right)\,.$$

Now we look for the dominant term in $\sum_{i=0}^{2^\rho-1}(i+z2^\rho)^2$, to further simplify the expression of $\mathbb{E}(X|Y=y)$

$$\begin{aligned}\sum_{i=0}^{2^\rho-1}(i+z2^\rho)^2 &= \sum_{i=0}^{2^\rho-1} i^2 + 2iz2^\rho + (z2^\rho)^2\,,\\ &= \sum_{i=0}^{2^\rho-1} i^2 + \sum_{i=0}^{2^\rho-1} 2iz2^\rho + (2^\rho-1)(z2^\rho)^2\,,\\ &\approx \frac{1}{3}2^{3\rho} + z2^{3\rho} + z^2 2^{3\rho}\,,\\ &\approx \left(z^2+z+\frac{1}{3}\right)2^{3\rho}\,.\end{aligned}$$

Putting these together, we get

$$\begin{aligned}\mathbb{E}(X) &= \sum_{z=0}^{2^l-1} \frac{p_\varepsilon^{\mathrm{HW}(z)}(1-p_\varepsilon)^{l-\mathrm{HW}(z)}}{2^\rho} \sum_{i=0}^{2^\rho-1}(i+z2^\rho)^2\,,\\ &\approx \sum_{z=0}^{2^l-1} p_\varepsilon^{\mathrm{HW}(z)}(1-p_\varepsilon)^{l-\mathrm{HW}(z)}\left(z^2+z+\frac{1}{3}\right)2^{2\rho}\,.\end{aligned}$$

And finally, we can express a bound on $\mathbb{E}(\|\boldsymbol{v}\|)$:

$$\mathbb{E}(\|\boldsymbol{v}\|) \leq \sqrt{(t+1)q^2\sum_{z=0}^{2^l-1} p_\varepsilon^{\mathrm{HW}(z)}(1-p_\varepsilon)^{l-\mathrm{HW}(z)}\left(z^2+z+\frac{1}{3}\right)2^{2\rho}}\,.$$

We can then confront the Gaussian heuristic with the expected norm of our target vector for different probabilities of error p_ε. This is illustrated in Figure 5 for an RSA-1024 modulus when the 16 MSB are known to be null.

We can see that even for moderate error probability, the expected norm hardly passes below the Gaussian heuristic. We confirmed in practice that attacks with the *scaling* approach do not outperform (in terms of success rate) the naive approach when all bit predictions have the same error probability.

5 Counter-Measures and Extensions

Several counter-measures have been proposed in the literature to protect RSA computations against fault and side-channel attacks. In this section, we review some of them in order to assess their resistance to our attack. Additionally, we extend our attack to RSA Blind Signatures in Section 5.4.

5.1 Message Randomization

Until very recently, encoding methods that introduce random salt in the message that is being processed by the signature generation operation were believed to be a sufficient protection against fault attacks. Indeed, to perform a fault attack like the Bellcore attack, one must be able to ask for the signature of the same encoded message twice, or at least to know the value of the message. This is required to compute either $\gcd(s-\tilde{s},N)$ or

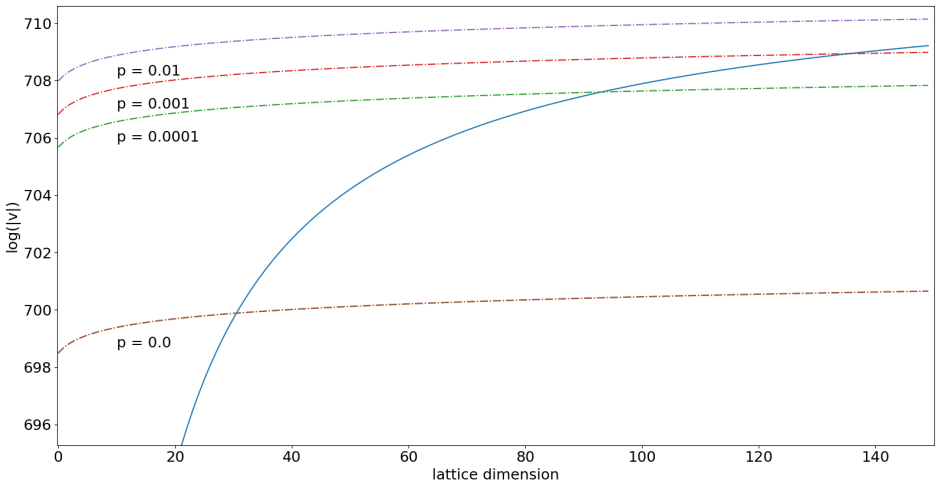


Figure 5: Evolution of the Gaussian heuristic for various lattice dimensions (plain) compared to the expected norm of the target vector (dashed) for different error probabilities in the assumed 16 MSBs.

$\gcd(m - \tilde{s}^e, N)$. By definition, this is not possible when the encoding introduces some random salt.

The attack introduced by [FGL+12], and presented here in Section 3, does not require to sign twice the same input or to know it, even partially. As a result, it is also applicable to randomized encodings such as PKCS#1 v1.5 [Kal98] and even PKCS#1 v2.2 (PSS) from [MKJR16].

5.2 Signature Verification

A common countermeasure to protect RSA-CRT implementations from fault attacks consists in verifying if the signature is valid before returning it. Indeed, this is not an expensive process as it just requires to elevate s to the power of the public exponent e modulo N and check if the result is equal to the message.

In the context of a fault attack forcing the l MSB of s_q at 0, it turns out that the signature might not be faulted with probability $1/2^l$. Indeed, it can happen that the l MSB of s_q *are actually* 0. This is called safe-error attack in the literature [YJ00].

In such cases, even if the signature is verified with the public exponent before being output, one out of 2^l signature is released to the attacker. As a result, one just needs to fault 2^l times more signatures to gather the sufficient number of signatures. Even if this complicate the attacker's task, we conclude that verifying might not be enough.

5.3 Message Blinding

Additive Message Blinding. A popular counter-measure to protect RSA-CRT implementations from side-channel attacks is *additive message blinding* [Cor99]. It may have a negative impact on the attack we present here. For the sake of clarity, we recall this masking scheme before stating how it affects the attack.

We consider that the message is blinded before the exponentiation modulo p' where p' is an extended modulus: $p' = \alpha \cdot p$ for a k-bit random value α. This is done by computing $m_p{}^\star = m + r_p \cdot p \bmod p'$, where r_p is a k-bit random mask. Then, $m_p{}^\star$ goes through the modular exponentiation modulo p' to obtain $s_p{}^\star = {m_p{}^\star}^{d_p} \bmod p'$. And finally, after doing likewise to compute $s_p{}^\star$, one can recover the signature as

$$s = ({s_q}^\star + q \cdot [({s_p}^\star - {s_q}^\star) \cdot i_q \bmod p']) \bmod N.$$

We can still see a relation of the form

$$s \cdot p = {s_q}^\star \cdot p + \lambda N.$$

However, in this case ${s_q}^\star$ is k bits larger than it is without the blinding scheme. As a result, the attacker needs to know k more bits of ${s_q}^\star$.

Multiplicative Message Blinding. Multiplicative message blinding [Koc96] is another blinding technique that can be found for instance in OpenSSL's implementation [Ope24]. According to this counter-measure, the message is again blinded before the modular exponentiations, but this time it is done by multiplying it by $r^e \bmod N$, with r a random value in $\mathbb{Z}_N^\star$. The modular exponentiation is then computed for $m' = m \cdot r^e \bmod N$ as usual

$$s' = s'_q + q \cdot [(s'_p - s'_q) \cdot i_q \bmod p],$$

and the signature is unmasked by multiplying it with the inverse of r. Indeed, we have

$$s' = {m'}^d = (m \cdot r^e)^d = m^d \cdot r \bmod N,$$

and so

$$s' \cdot r^{-1} = m^d = s \bmod N.$$

In this case, the faulty signature would still be blinded somehow, as $\tilde{s} = \tilde{s'} \cdot r^{-1} \bmod N$, and without the knowledge of r we cannot go any further.

Montgomery representation. In 1985, Montgomery introduced a *Modular multiplication without trial division* [Mon85]. This method is efficient when the modulus has no particular form, which is typically the case for RSA and has been widely adopted in various hardware public-key accelerators and cryptographic libraries [Ope24].

Letting the Montgomery radix $R = r^n = 2^{wn}$ (adapted for handling integer of n w-bit words), the Montgomery multiplication takes as input A' and B' (the Montgomery representation of A and B such that $X' = X \cdot R \bmod N$) and returns

$$M(A', B', N) = A' \cdot B' \cdot R^{-1} = C' \bmod N\,.$$

We can note that the Montgomery representation of X can be obtained by computing $M(X, R^2, N)$, and the way back to the regular representation by $M(\tilde{X}, 1, N)$, and refer the reader to [BL17] for a deeper view on Montgomery multiplication.

In our case, we can then expect that the modular operations are performed in the Montgomery representation. It is thus natural to question the validity of our attack in this context, as the intermediate result we want to fault is actually

$$s'_q = s_q \cdot R \bmod q\,.$$

In [FGL+12], the authors have exploited faults during the modular exponentiation using Montgomery arithmetic. In our case, we rather target the recombination of the two sub-exponentiations, and we can note that Garner's formula does not use modular reductions. We can then expect that it is processed with the regular representation. However, if the masking scheme is preserved to protect the recombination a final modular reduction is required and it is then natural to question the behaviour of our attack if the Montgomery representation is used.

In the case of the multiplicative blinding, the conclusion remains that the mask prevents the attack success. In the case of the additive blinding, the faulty signature is then

$$s' = R\,(s_q{}^\star + q \cdot [(s_p{}^\star - s_q{}^\star) \cdot i_q \bmod p']) \bmod N.$$

To return to the regular representation, one must multiply s' by R^{-1} and reduce the result modulo N. Even with the knowledge of the Montgomery constant R, this results in the lost of any information on the value of $s \bmod q$.

5.4 RSA Blind Signatures

In a Blind Signature scheme, a client obtains a signature s on a message m from a server owning the secret key, in such a way that the server learns nothing about the signed message m.

In the current draft for RSA Blind Signatures [DJW23], the client sends the blinded message m' for signature to a remote server $m' = m \cdot r^{-e} \bmod N$ for a random value $r \leftarrow \mathbb{Z}_N^\star$. Then the server computes and returns the blinded signature $s' = s'_q + q \cdot [(s'_p - s'_q) i_q \bmod p]$. We can see that if s' and some information on s'_q are available to an adversary, a key recovery can be settled.

Such an attack might be difficult to mount in practice. Moreover, [DJW23] mandates that the server verifies ${s'}^e = m'$ before sending back s'. However, one cannot consider that faulty signatures will never happen. For instance, [Wei15, SSHW22] observe that an attack against TLS can be mount simply by connecting to a server and waiting that a fault occurs during the computation. A realistic attack against SSH and IPsec based on the same idea is mounted in [RHSH23].

6 Conclusion

In this work, we revisited an attack against RSA-CRT signatures, using a reduction of the originally formulated problem and recent advances in lattice reductions. Following our approach, the required number of known bits drops from 32 down to 7. Besides, we have shown that larger instances of RSA can also be targeted by the PACD-based attack and extended the analysis of the error tolerance of the BDD with Predicate approach. Analysing the effect of standard countermeasures, we stress the fact that a verification of the signature is not a sufficient protection. Message blinding with k-bit masks appears as an efficient countermeasure when k is large enough. Additionally, we showed that the Partial ACD problem may be solved by a reduction to the HNP problem, which does not seem to have been noticed yet in the literature and might be of independent interest.

References

[ABF+03] Christian Aumüller, Peter Bier, Wieland Fischer, Peter Hofreiter, and Jean-Pierre Seifert. Fault attacks on RSA with CRT: Concrete results and practical countermeasures. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES 2002*, volume 2523 of *LNCS*, pages 260–275. Springer, August 2003. doi:10.1007/3-540-36400-5_20.

[ADH+19] Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens. The general sieve kernel and new records in lattice reduction. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 717–746. Springer, May 2019. doi:10.1007/978-3-030-17656-3_25.

[AH21] Martin R. Albrecht and Nadia Heninger. On bounded distance decoding with predicate: Breaking the "lattice barrier" for the hidden number problem. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 528–558. Springer, October 2021. doi:10.1007/978-3-030-77870-5_19.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996. doi:10.1145/237814.237838.

[BCG08] Alexandre Berzati, Cécile Canovas, and Louis Goubin. In(security) Against Fault Injection Attacks for CRT-RSA Implementations. In Luca Breveglieri, Shay Gueron, Israel Koren, David Naccache, and Jean-Pierre Seifert, editors, *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2008*, pages 101–107. IEEE Computer Society, 2008. doi:10.1109/FDTC.2008.9.

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In Robert Krauthgamer, editor, *27th SODA*, pages 10–24. ACM-SIAM, January 2016. doi:10.1137/1.9781611974331.ch2.

[BDL01] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of eliminating errors in cryptographic computations. *Journal of Cryptology*, 14(2):101–119, March 2001. doi:10.1007/s001450010016.

[Bel96] Bellcore. New Threat Model Breaks Crypto Codes. Press Release, September 1996. URL: https://pi4.informatik.uni-mannheim.de/pi4.data/content/projects/cryptography/rgp/dfa/facts.html.

[BL17] Joppe W. Bos and Arjen K. Lenstra, editors. *Topics in Computational Number Theory Inspired by Peter L. Montgomery*. Cambridge University Press, 2017. doi:10.1017/9781316271575.

[BS97] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 513–525. Springer, August 1997. doi:10.1007/BFb0052259.

[CH13] Henry Cohn and Nadia Heninger. Approximate common divisors via lattices. In Everett W. Howe and Kiran S. Kedlaya, editors, *Algorithmic Number Theory – ANTS X*, volume 1 of *The Open Book Series*, pages 271–293, 2013. doi:10.2140/obs.2013.1.271.

[CJ05] Mathieu Ciet and Marc Joye. Elliptic curve cryptosystems in the presence of permanent and transient faults. *DCC*, 36(1):33–43, 2005. doi:10.1007/s10623-003-1160-8.

[CJK+09] Jean-Sébastien Coron, Antoine Joux, Ilya Kizhvatov, David Naccache, and Pascal Paillier. Fault attacks on RSA signatures with partially unknown messages. In Christophe Clavier and Kris Gaj, editors, *CHES 2009*, volume 5747 of *LNCS*, pages 444–456. Springer, September 2009. doi:10.1007/978-3-642-04138-9_31.

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, December 2011. doi:10.1007/978-3-642-25385-0_1.

[Cor99] Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Çetin Kaya Koç and Christof Paar, editors, *CHES'99*, volume 1717 of *LNCS*, pages 292–302. Springer, August 1999. doi:10.1007/3-540-48059-5_25.

[DJW23] Frank Denis, Frederic Jacobs, and Christopher A. Wood. RSA Blind Signatures. RFC 9474, October 2023. doi:10.17487/RFC9474.

[DLV03] Pierre Dusart, Gilles Letourneux, and Olivier Vivolo. Differential fault analysis on AES. In Jianying Zhou, Moti Yung, and Yongfei Han, editors, *ACNS 03*, volume 2846 of *LNCS*, pages 293–306. Springer, October 2003. doi:10.1007/978-3-540-45203-4_23.

[dt23] The FPLLL development team. fplll, a lattice reduction library, Version: 5.4.5, 2023. URL: https://github.com/fplll/fplll.

[EL10] Nevine Maurice Ebeid and Rob Lambert. A new CRT-RSA algorithm resistant to powerful fault attacks. In *Workshop on Embedded Systems Security – WESS 2010*, pages 1–8. ACM, 2010. doi:10.1145/1873548.1873556.

[FGL+12] Pierre-Alain Fouque, Nicolas Guillermin, Delphine Leresteux, Mehdi Tibouchi, and Jean-Christophe Zapalowicz. Attacking RSA-CRT signatures with faults on Montgomery multiplication. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 447–462. Springer, September 2012. doi:10.1007/978-3-642-33027-8_26.

[GGM16] Steven D. Galbraith, Shishay W. Gebregiyorgis, and Sean Murphy. Algorithms for the approximate common divisor problem. *LMS Journal of Computation and Mathematics*, 19(A):58–72, 2016. doi:10.1112/S1461157016000218.

[Gir06] Christophe Giraud. An RSA Implementation Resistant to Fault Attacks and to Simple Power Analysis. *IEEE Transactions on Computers*, 55(9):1116–1120, 2006. doi:10.1109/TC.2006.135.

[GN08] Nicolas Gama and Phong Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 207–216. ACM Press, May 2008. doi:10.1145/1374376.1374408.

[How01] Nick Howgrave-Graham. Approximate Integer Common Divisors. In Joseph H. Silverman, editor, *Cryptography and Lattices – CaLC'01*, volume 2146 of *LNCS*, pages 51–66. Springer, 2001. doi:10.1007/3-540-44670-2_6.

[HS07] Guillaume Hanrot and Damien Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 170–186. Springer, August 2007. doi:10.1007/978-3-540-74143-5_10.

[JQBD97] Marc Joye, Jean-Jacques Quisquater, Feng Bao, and Robert H. Deng. RSA-type signatures in the presence of transient faults. In Michael Darnell, editor, *Cryptography and Coding*, volume 1355 of *LNCS*, pages 155–160. Springer, 1997. doi:10.1007/BFb0024460.

[Kal98] Burt Kaliski. PKCS#1: RSA Encryption Version 1.5. RFC 2313, March 1998. doi:10.17487/RFC2313.

[Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 104–113. Springer, August 1996. doi:10.1007/3-540-68697-5_9.

[KQ07] Chong Hee Kim and Jean-Jacques Quisquater. Fault attacks for CRT based RSA: New attacks, new results, and new countermeasures. In Damien Sauveron, Konstantinos Markantonakis, Angelos Bilas, and Jean-Jacques Quisquater, editors, *IFIP International Workshop on Information Security Theory and Practices – WISTP'07*, volume 4462 of *LNCS*, pages 215–228. Springer, 2007. doi:10.1007/978-3-540-72354-7_18.

[Len96] Arjen K. Lenstra. Memo on RSA signature generation in the presence of faults. Technical Report EPFL, 1996. URL: http://infoscience.epfl.ch/record/164524.

[LFZD16] Pei Luo, Yunsi Fei, Liwei Zhang, and A. Adam Ding. Differential Fault Analysis of SHA3-224 and SHA3-256. In *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2016*, pages 4–15. IEEE Computer Society, 2016. doi:10.1109/FDTC.2016.17.

[LLL82] Arjen K. Lenstra, Hendrik W. Lesntra, and László Lovász. Factoring polynomials with rational coefficients. *Matematische Annalen*, 261:366–389, 1982. doi:10.1007/BF01457454.

[MKJR16] Kathleen Moriarty, Burt Kaliski, Jakob Jonsson, and Andreas Rusch. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.2. RFC 8017, November 2016. doi:10.17487/RFC8017.

[Mon85] Peter L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44(170):519–521, 1985. doi:10.1090/S0025-5718-1985-0777282-X.

[MW16] Daniele Micciancio and Michael Walter. Practical, predictable lattice basis reduction. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 820–849. Springer, May 2016. doi:10.1007/978-3-662-49890-3_31.

[NS01] Phong Q. Nguyen and Jacques Stern. The Two Faces of Lattices in Cryptology. In Joseph H. Silverman, editor, *Cryptography and Lattices – CaLC'01*, volume 2146 of *LNCS*, pages 146–180. Springer, 2001. doi:10.1007/3-540-44670-2_12.

[NS06] Phong Q. Nguyen and Damien Stehlé. LLL on the average. In Florian Hess, Sebastian Pauli, and Michael E. Pohst, editors, *Algorithmic Number Theory – ANTS-VII*, volume 4076 of *LNCS*, pages 238–256. Springer, 2006. doi:10.1007/11792086_18.

[NS09] Phong Q. Nguyen and Damien Stehlé. An LLL Algorithm with Quadratic Complexity. *SIAM Journal on Computing*, 39(3):874–903, 2009. doi:10.1137/070705702.

[Ope24] OpenSSL. Cryptography and SSL/TLS toolkit, 2024. URL: https://www.openssl.org/.

[PQ03] Gilles Piret and Jean-Jacques Quisquater. A differential fault attack technique against SPN structures, with application to the AES and KHAZAD. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *CHES 2003*, volume 2779 of *LNCS*, pages 77–88. Springer, September 2003. doi:10.1007/978-3-540-45238-6_7.

[QC82] Jean-Jacques Quisquater and Chantal Couvreur. Fast decipherment algorithm for RSA public-key cryptosystem. *Electronics Letters*, 21(18):905–907, 1982. doi:10.1049/el:19820617.

[RG14] Pablo Rauzy and Sylvain Guilley. Countermeasures against High-Order Fault-Injection Attacks on CRT-RSA. In Assia Tria and Dooho Choi, editors, *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2014*, pages 68–82. IEEE Computer Society, 2014. doi:10.1109/FDTC.2014.17.

[RH23] Keegan Ryan and Nadia Heninger. Fast practical lattice reduction through iterated compression. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 3–36. Springer, August 2023. doi:10.1007/978-3-031-38548-3_1.

[RHSH23] Keegan Ryan, Kaiwen He, George Arnold Sullivan, and Nadia Heninger. Passive SSH key compromise via lattices. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023*, pages 2886–2900. ACM Press, November 2023. doi:10.1145/3576915.3616629.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. doi:10.1145/359340.359342.

[Rya23] Keegan Ryan. flatter, a (f)ast (lat)tice (r)eduction library, 2023. URL: https://github.com/keeganryan/flatter.

[Sch87] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53:201–224, 1987. doi:10.1016/0304-3975(87)90064-8.

[SE94] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66:181–199, 1994. doi:10.1007/BF01581144.

[She21] Laura Shea. Generalize ECDSA solver to work for arbitrary MSB/LSB, 2021. URL: https://github.com/malb/bdd-predicate/pull/6.

[SSHW22] George Arnold Sullivan, Jackson Sippe, Nadia Heninger, and Eric Wustrow. Open to a fault: On the passive compromise of TLS keys via transient errors. In Kevin R. B. Butler and Kurt Thomas, editors, *USENIX Security 2022*, pages 233–250. USENIX Association, August 2022.

[Vig08] David Vigilant. RSA with CRT: A new cost-effective solution to thwart fault attacks. In Elisabeth Oswald and Pankaj Rohatgi, editors, *CHES 2008*, volume 5154 of *LNCS*, pages 130–145. Springer, August 2008. doi:10.1007/978-3-540-85053-3_9.

[Wei15] Florian Weimer. Factoring RSA Keys With TLS Perfect Forward Secrecy. Technical Report Red Hat, 2015. URL: https://people.redhat.com/~fweimer/rsa-crt-leaks.pdf.

[YJ00] Sung-Ming Yen and Marc Joye. Checking before output may not be enough against fault-based cryptanalysis. *IEEE Transactions on Computers*, 49(9):967–970, 2000. doi:10.1109/12.869328.

[YKM06] Sung-Ming Yen, Dongryeol Kim, and Sang-Jae Moon. Cryptanalysis of Two Protocols for RSA with CRT Based on Fault Infection. In Luca Breveglieri, Israel Koren, David Naccache, and Jean-Pierre Seifert, editors, *Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2006*, volume 4236 of *LNCS*, pages 53–61. Springer, 2006. doi:10.1007/11889700_5.