New Results for Partial Key Exposure on RSA with Exponent Blinding

Stelvio Cimato[1], Silvia Mella[1] and Ruggero Susella[2]
[1]*Università degli Studi di Milano, Milano, Italy*
[2]*STMicroelectronics, Agrate Brianza, Italy*

Keywords: RSA, Partial Key Exposure, Coppersmith's Method, Exponent Blinding, Horizontal Attack.

Abstract: In 1998, Boneh, Durfee and Frankel introduced partial key exposure attacks, a novel application of Coppersmith's method, to retrieve an RSA private key given only a fraction of its bits. This type of attacks is of particular interest in the context of side-channel attacks. By applying the exponent blinding technique as a countermeasure for side-channel attacks, the private exponent becomes randomized at each execution. Thus the attacker has to rely only on a single trace, significantly incrementing the noise, making the exponent bits recovery less effective. This countermeasure has also the side-effect of modifying the RSA equation used by partial key exposure attacks, in a way studied by Joye and Lepoint in 2012. We improve their results by providing a simpler technique in the case of known least significant bits and a better bound for the known most significant bits case. Additionally, we apply partial key exposure attacks to CRT-RSA when exponent blinding is used, a case not yet analyzed in literature. Our findings, for which we provide theoretical and experimental results, aim to reduce the number of bits to be recovered through side-channel attacks in order to factor an RSA modulus when the implementation is protected by exponent blinding.

1 INTRODUCTION

At Eurocrypt 1996 Don Coppersmith presented a novel method to find small solutions of univariate modular polynomials, with some applications to the RSA cryptosystem (Coppersmith, 1996b). He extended the method to bivariate equations, which allowed to factor an RSA modulus given half of the bits of one of its prime factors (Coppersmith, 1996a).

In 1998, Boneh, Durfee and Frankel introduced partial key exposure, a family of attacks on RSA requiring the knowledge of some consecutive most significant bits (MSB) or least significant bits (LSB) of the private exponent (Boneh et al., 1998). The main idea behind their methods, for the common cases where the factorization of the public exponent is known, is to use the given partial information on the private exponent to obtain partial information on a prime factor of the modulus, and then apply Coppersmith's method to factor.

This application is of high interest in the context of side-channel attacks. Side-Channel attacks, introduced in 1996 by Paul Kocher (Kocher, 1996), are attacks on physical implementations of cryptographic algorithms. In these attacks a side-channel information of the computation (such as power consumption, electromagnetic emission, acoustic emission, etc.) is used to recover the secret used during the computation.

Most side channel attacks leverage on combining the side-channel leakages, i.e. traces, of several executions of the cryptographic algorithm with same secret but different input. The first attack of this family is Differential Power Analysis (DPA) (Kocher et al., 1999). Its main feature is the ability to significantly reduce the random noise, by averaging a large amount of traces, compared to Simple Power Analysis (SPA), where only one trace is used.

The reader may now wonder why an attacker might be able to obtain information on a part of the secret exponent and not on the entire exponent. The reason is that some countermeasures can be adopted by implementers to thwart side channel attacks.

A common countermeasure used for RSA is exponent blinding, originally introduced in (Kocher, 1996) but often attributed to (Coron, 1999). It consists of adding a random multiple of $\phi(N)$ to the RSA private exponent at each execution. This countermeasure has the feature to change the private exponent at each computation, thus not permitting the use of multiple traces, as required for DPA. This results in the need of using a single trace to discover the secret key.

Cimato S., Mella S. and Susella R..
New Results for Partial Key Exposure on RSA with Exponent Blinding.
DOI: 10.5220/0005571701360147
In *Proceedings of the 12th International Conference on Security and Cryptography* (SECRYPT-2015), pages 136-147
ISBN: 978-989-758-117-5

A method for this was originally proposed in (Walter, 2001), for a particular exponentiation algorithm, and generalized for regular exponentiation algorithms in (Clavier et al., 2010) and named horizontal attack.

The practical issue that the adversary faces in the horizontal setting is the high noise. Therefore, the adversary is able to correctly guess only a partial number of the bits of the exponent and not the entire exponent.

The application of partial key exposure, when exponent blinding is used as side-channel countermeasure, would allow the imperfect attacker to still recover the correct private exponent. However, the equation exploited by Boneh, Durfee and Frankel is modified by the introduction of exponent blinding and their attacks don't apply anymore.

The question as to whether partial key exposure could be applied in this setting was answered in (Joye and Lepoint, 2012). The authors presented two techniques to recover the full exponent, knowing enough MSB or LSB portions of it, leaving the open question as up to which extent it is possible to apply partial key exposure when exponent blinding is applied to the CRT variant of RSA (Quisquater and Couvreur, 1982).

Our contribution consists of new methods for partial key exposure when exponent blinding is used, improving the results of (Joye and Lepoint, 2012) for common RSA settings and providing novel attacks for the CRT variant. Specifically, in this work, we:

- reduce the number of required bits for the MSB attack and make it to not rely on a common heuristic assumption;
- provide a more efficient technique for the LSB attack, requiring to reduce a lattice basis of lower dimension;
- present novel attacks against CRT-RSA implementations that make use of exponent blinding. This particular case has never been analyzed before.

This work is organized as follows. In Section 2 we recall some basic information about RSA. In Section 3 we give a brief introduction about lattices and Coppersmith's method. In Section 4 we present two partial key exposure attacks on RSA with exponent blinding and in Section 5 on CRT-RSA with exponent blinding. Experimental results are then provided in Section 6.

2 RSA APPLICATIONS

In literature, Coppersmith's method has been applied with very different, and unusual, RSA parameters. For example the case where e is of the same bitsize of N has been analyzed in (Ernst et al., 2005). In this work we preferred to focus our analysis on more common RSA settings.

Let (N,e) be a RSA public key. The modulus $N = pq$ has prime factors p and q of equal bit-size. We assume wlog that $p > q$, that implies

$$q < \sqrt{N} < p < 2q < 2\sqrt{N}$$

and

$$\sqrt{N} < p+q < 3\sqrt{N}.$$

It is common practice to choose 1024 or 2048-bit modulus N.

The most common value for the public exponent e is $2^{16}+1$. This is also the default value for the public exponent in the OpenSSL library. Other common values are 3 and 17. NIST mandates that e satisfies $2^{16} < e < 2^{256}$ (Kerry et al., 2013). Therefore, to be as generic as possible but still adhering to realistic scenarios, we will consider in our analysis $3 \leq e < 2^{256}$, but we will provide experiments only for the most common case $e = 2^{16}+1$.

The private exponent d satisfies $ed-1 = k\phi(N)$ for some integer k, where $\phi(N)$ denotes the Euler totient function. The exponent d is commonly chosen to be full size, namely as large as $\phi(N)$. In order to speed-up the decryption process, someone suggests to use smaller d. However, this choice may lead to security problems as Wiener's attack (Wiener, 1990). Therefore, it is usually avoided.

The side-channel countermeasure considered in this work is the exponent blinding introduced by Kocher (Kocher, 1996). It consists of adding a random multiple of $\phi(N)$ to d, thus RSA exponentiation is computed by using the new exponent $d^* = d + \ell\phi(N)$, for some $\ell > 0$. The dimension of ℓ is a tradeoff between security and efficiency. If ℓ is 32-bit long or smaller, it allows some combination of brute-forcing and side-channel as in (Fouque et al., 2006), where a brute-force on ℓ is required. Thus, it is a safer choice to use ℓ with bit-size 64. A larger dimension would make the decryption process less efficient.

In our analysis, to maintain generality, we will consider $0 \leq \ell < 2^{128}$ and in our experiments we will test bit-sizes of 0, 10, 32, 64 and 100. Our methods never require the capability of brute-forcing the values of k or ℓ, sometimes needed in other works.

In order to speed up the exponentiation computation, some RSA implementations make use of a technique based on the Chinese Remainder Theorem (CRT). In particular, one can use exponents

$$d_p = d \bmod (p-1) \quad \text{and} \quad d_q = d \bmod (q-1)$$

to compute

$$x^{d_p} \bmod p \quad \text{and} \quad x^{d_q} \bmod q.$$

Then, the two results can be combined using the CRT to obtain $x^d \bmod N$ (Quisquater and Couvreur, 1982).

Also CRT-RSA can be protected with exponent blinding. Thus, exponentiation is computed by using $d_p^* = d_p + \ell_1(p-1)$ and $d_q^* = d_q + \ell_2(q-1)$, for some $\ell_1, \ell_2 > 0$.

In this work we will consider both RSA and CRT-RSA implementations that make use of the exponent blinding countermeasure. Our RSA settings will consider moduli of 1024 and 2048 bits, public exponent such that $3 \leq e < 2^{256}$, private exponent of full size and a randomization factor up to 128 bits.

To derive theoretical bounds in next sections, we prefer to express the restrictions on e and ℓ with respect to the modulus N. In general, we translate them to the less restrictive conditions: $\ell < 2N^{\frac{1}{8}}$ and $e < 2N^{\frac{1}{4}}$. When necessary, we will consider more restrictive bounds.

We will run experiments by considering the widely used public exponent $e = 2^{16}+1$ and random values ℓ of different bit-size from 0 to 100. The modulus N will be 2048-bit long, but note that our attacks are effective also for other sizes.

3 GENERAL STRATEGY

Partial key exposure attacks relies on Coppersmith's method for finding roots of modular polynomials and multivariate polynomials. This method makes significant use of lattices and lattice reduction algorithms.

We give here a brief introduction to lattices and to the general strategy used in partial key exposure attacks and thus also in our attacks.

3.1 Lattices

Given a set of real linearly independent vectors $B = \{b_1, \ldots, b_n\}$ with $b_i \in \mathbb{R}^n$, a (full-rank) lattice spanned by B is the set of all integer linear combinations of vectors of B. Namely, the set $L(B) = \{\sum_i x_i b_i \,:\, x_i \in \mathbb{Z}\}$.

B is called the *basis* of the lattice and the $(n \times n)$-matrix consisting of the row vectors $b_1, \ldots, b_n$ is called basis matrix.

Every lattice has an infinite number of lattice bases. A basis is obtained from another through a unimodular transformation (i.e., by multiplying the basis matrix by a matrix with determinant ± 1). The determinant of the lattice is defined as $\det(L) = |\det(B_i)|$ and is an invariant, namely it is independent of the choice of the basis. The dimension of the lattice is $\dim(L) = n$.

The goal of lattice reduction is to find a basis with short and nearly orthogonal vectors. The LLL algorithm (Lenstra et al., 1982) produces in polynomial time a set of reduced basis vectors. The following theorem bounds the norm of these vectors.

Theorem 1 (Lenstra-Lenstra-Lovász)**.** *Let L be a lattice of dimension n. The LLL-algorithm outputs in polynomial time reduced basis vectors v_i, $1 \leq i \leq n$, satisfying*

$$\|v_1\| \leq \|v_2\| \leq \ldots \leq \|v_i\| \leq 2^{\frac{n(n-1)}{4(n+1-i)}} \det L^{\frac{1}{n+1-i}}$$

3.2 General Strategy

In (Coppersmith, 1996b), Don Coppersmith presents a rigorous method to find small roots of univariate modular polynomials. The method is based on LLL and can be extended to polynomials in more variables, but only heuristically.

In this work we use the following reformulation of Coppersmith's theorem due to Howgrave-Graham (Howgrave-Graham, 1997).

Theorem 2 (Howgrave-Graham)**.** *Let $f(x_1, \ldots, x_k)$ be a polynomial in k variables with n monomials. Let m be a positive integer. Suppose that*

1. $f(r_1, \ldots, r_k) = 0 \bmod b^m$ *where* $|r_i| < X_i \; \forall i$
2. $\|f(x_1X_1, \ldots, x_kX_k)\| < \dfrac{b^m}{\sqrt{n}}$

Then $f(r_1, \ldots, r_k) = 0$ holds over the integers.

The general strategy is the following. Starting from an RSA equation we construct a multivariate polynomial $f_b(x_1, \ldots, x_k)$ modulo an integer b, such that its root $(r_1, \ldots, r_k)$ contains secret values. Our goal is to find this root, even if no classic root finding method is known for modular polynomials. So, we construct k polynomials $f_1, \ldots, f_k$ satisfying the two conditions of Theorem 2 so that such polynomials will have the same root $(r_1, \ldots, r_k)$ over $\mathbb{Z}$. Finally, we compute the common roots of these polynomials and recover the secret values.

To generate such polynomials we apply the following strategy. Starting from f_b we construct auxiliary polynomials $g_i(x_1, \ldots, x_k)$ that all satisfy condition 1 of Howgrave-Graham's Theorem. Since every integer linear combination of these polynomials also satisfies condition 1, we look for linear combinations that also satisfy condition 2. Such combinations are the polynomials $f_1, \ldots, f_k$.

In order to construct $f_1, \ldots, f_k$, we build a lattice $L(B)$ where the basis B is composed by the coefficient vectors of the polynomials $g_i(x_1X_1, \ldots, x_kX_k)$ (with $X_1, \ldots, X_k$ bounds on the root as in Theorem 2).

By using the LLL-lattice reduction algorithm, we obtain a reduced basis for the lattice L as in Theorem 1. The first k vectors of the reduced basis have norm smaller than $\frac{b^m}{\sqrt{n}}$, if:

$$2^{\frac{n(n-1)}{4(n+1-k)}} \det L^{\frac{1}{n+1-k}} < \frac{b^m}{\sqrt{n}}$$

We may let terms that do not depend on N contribute to an error term ε and consider the simplified condition

$$\det L \leq b^{m(n+1-k)}. \qquad (1)$$

If this condition holds, then we can use the first k reduced-basis vectors to construct the polynomials $f_1, \ldots, f_k$ satisfying the second condition of Theorem 2. Then, in order to compute $(r_1, \ldots, r_k)$, we do the following.

If $k = 1$, then we consider the polynomial $F = f_1(x_1)$ and apply a classic roots finding algorithm for univariate polynomials over the integers.

If $k > 1$, we use the resultant computation to construct k univariate polynomials $F_i(x_i)$ from $f_1, \ldots, f_k$ and apply a classic roots finding algorithm for each of them. The effectiveness of this last method relies on the following heuristic assumption.

Assumption 1. *The resultant computation for the polynomials f_i described above yields a non-zero polynomial.*

This assumption is fundamental and widely used for many attacks in literature (Joye and Lepoint, 2012; Lu et al., 2014; Blömer and May, 2003; Boneh et al., 1998; Ernst et al., 2005). None of our experiments has ever failed to yield a non-zero polynomial and hence to mount the attack.

In this work we will make use of a seminal result due to Coppersmith, based on the strategy described above. We present here a more general variant of it, due to May (May, 2003), together with a sketch of its proof to illustrate how we will construct lattices for our experiments.

Theorem 3. *Let $N = pq$ with $p > q$. Let k be an unknown integer that is not a multiple of q. Suppose we know an approximation $\widetilde{kp}$ of kp with $|kp - \widetilde{kp}| \leq 2N^{\frac{1}{4}}$. Then we can factor N in time polynomial in $\log N$.*

Sketch of proof. Define the univariate polynomial

$$f_p(x) = x + \widetilde{kp}$$

with root $x_0 = kp - \widetilde{kp}$ modulo p.
Divide the interval $[-2N^{\frac{1}{4}}, 2N^{\frac{1}{4}}]$ into 8 subintervals of size $\frac{1}{2}N^{\frac{1}{4}}$ centered at some x_i.
For each subinterval consider the polynomial $f_p(x - x_i)$ and find its roots r such that $|r| \leq \frac{1}{4}N^{\frac{1}{4}}$. Among all these roots of all these polynomials there is also x_0.
So, for each $f_p(x - x_i)$ set $X = \frac{1}{4}N^{\frac{1}{4}}$. Fix $m = \lceil \log N/4 \rceil$ and set $t = m$.
Define the auxiliary polynomials

$$g_{i,j}(x) = x^j N^i f^{m-i} \text{ for } i = 0, \ldots, m-1;\ j = 0$$
$$h_i(x) = x^i f^m(x) \text{ for } i = 0, \ldots, t-1$$

and construct the lattice spanned by the vectors $g_{i,j}(xX)$ and $h_i(xX)$.
By applying the LLL-algorithm to L, a reduced basis is obtained. From the shortest vector construct the polynomial $f_i(x)$. Among its roots over the integers, there are also the roots of $f_p(x - x_i)$. Compute the roots of $f_i(x)$ by using a classic roots-finding algorithm.
Construct the set R of all integer roots of the polynomials $f_i(x)$. The set R will contain also the root x_0.
Thus, $f(x_0) = kp$ can be computed and, since k is not a multiple of q, the computation of $\gcd(N, kp)$ gives p.

Recall that the LLL-algorithm is polynomial in the dimension of the matrix basis and in the bit-size of its entries. Since the dimension of the lattice is $m + t = \lceil \log N/2 \rceil$ and the bit-size of its entries is bounded by a polynomial in $(m \log N)$, every step of the proof can be done in polynomial time. □

4 ATTACKS ON RSA

In this section we present two attacks on RSA implementations, one given the most significant bits of the private exponent and the other one given its last significant bits. We assume that the private exponent d is full-size and that it is masked by a random multiple ℓ of $\phi(N)$. Thus, exponentiation is performed by using the exponent $d^* = d + \ell\phi(N)$ for some $\ell \geq 0$. When $\ell = 0$ clearly $d^* = d$, that means that no countermeasure is applied.

Joye and Lepoint presented partial key exposure attacks on RSA with exponent blinding (Joye and Lepoint, 2012). Here we present two alternative approaches that allow us to get better bounds on the number of leaked bits necessary to the attacker to break the system.

4.1 Partial Information on LSB of d^*

In this section, we assume that the attacker is able to recover the least significant bits of the secret d^*. We write $d^* = d_1 \cdot M + d_0$, where d_0 represents the fraction of d^* known to the attacker while d_1 represents the unknown part. For instance, if the attacker knows the m LSB of d^*, then $M = 2^m$.

We prove that if the attacker knows a sufficiently large number of least significant bits, then she can factor N.

To prove our result, we generalize the method used in (Blömer and May, 2003), by introducing the new factor ℓ.

Theorem 4. *Let (N,e) be an RSA public key with $e = N^{\alpha} \leq 2N^{\frac{1}{4}}$ and $d^* = d + \ell\phi(N)$, for some $\ell = N^{\sigma} \leq 2N^{\frac{1}{8}}$. Suppose we are given d_0 and M satisfying $d_0 = d^* \mod M$ with*

$$M \geq N^{\frac{1}{3}\sqrt{1+6(\alpha+\sigma)}+\frac{1}{6}(1+6\sigma)+\varepsilon},$$

for some $\varepsilon > 0$. Then, under Assumption 1, we can find the factorization of N in time polynomial in $\log N$.

Proof. We start from the RSA equation

$$ed - 1 = k\phi(N).$$

Since $d^* = d + \ell\phi(N)$, we obtain the equation

$$ed^* - 1 = (k + e\ell)\phi(N).$$

Let $k^* = k + e\ell$, so that $ed^* - 1 = k^*\phi(N)$.
By writing $d^* = d_1 M + d_0$ and considering that $\phi(N) = N - (p + q - 1)$, we get

$$k^*N - k^*(p+q-1) - ed_0 + 1 = eMd_1.$$

It follows that the bivariate polynomial

$$f_{eM}(x,y) = xN - xy - ed_0 + 1$$

has root $(x_0, y_0) = (k^*, p+q-1)$ modulo eM.
In order to bound x_0, notice that

$$k^* = \frac{ed^*-1}{\phi(N)} < e\left(\frac{d+\ell\phi(N)}{\phi(N)}\right) < e(1+\ell) \leq 2N^{\alpha+\sigma}.$$

In addition, recall that $p + q \leq 3N^{\frac{1}{2}}$.
We can set the bounds $X = 2N^{\alpha+\sigma}$ and $Y = 3N^{\frac{1}{2}}$ so that $x_0 \leq X$ and $y_0 \leq Y$.
To construct the lattice, we consider the following auxiliary polynomials

$$g_{i,j}(x,y) = x^i (eM)^i f_{eM}^{m-i} \text{ for } i = 0,\ldots,m;\ j = 0,\ldots,i$$
$$h_{i,j}(x,y) = y^j (eM)^i f_{eM}^{m-i} \text{ for } i = 0,\ldots,m;\ j = 1,\ldots,t$$

for some integers m and t, where $t = \tau m$ has to be optimized.
All integer linear combinations of these polynomials have the root (x_0, y_0) modulo $(eM)^m$, since they all have a term $(eM)^i f_{eM}^{m-i}$. So the first condition of Theorem 2 is satisfied. In order to satisfy the second condition, we have to find a short vector in the lattice spanned by $g_{i,j}(xX,yY)$ and $h_{i,j}(xX,yY)$. In particular, this vector shall have a norm smaller than $\frac{(eM)^m}{\sqrt{\dim L}}$. The second condition of Theorem 2 is satisfied when inequality (1) holds, i.e. if

$$\det L \leq (eM)^{m(n-1)}. \tag{2}$$

An easy computation shows that $n = \left(\tau + \frac{1}{2}\right) m^2$ and that

$$\det L(M) = \left((eMY)^{3\tau+2} Z^{3\tau^2+3\tau+1}\right)^{\frac{1}{6}m^3(1+o(1))}.$$

Considering the bounds $X = 2N^{\alpha+\sigma}$ and $Y = 3N^{\frac{1}{2}}$, we obtain the condition

$$\left((eM2N^{\alpha+\sigma})^{3\tau+2}(3N^{\frac{1}{2}})^{3\tau^2+3\tau+1}\right)^{\frac{1}{6}m^3(1+o(1))} \leq (eM)^{m(n-1)}$$

that reduces to

$$N^{\frac{m^3}{6}\left((\alpha+\sigma)(3\tau+2)+\frac{1}{2}(3\tau^2+3\tau+1)\right)(1+o(1))}$$
$$\leq (eM)^{m(n-1)-\frac{m^3}{6}(3\tau+2)(1+o(1))}.$$

We know that $eM \geq N^{\alpha\frac{1}{3}\sqrt{1+6(\alpha+\sigma)}+\frac{1}{6}(1+6\sigma)+\varepsilon}$, so the above condition is satisfied if

$$9\tau^2 + 6(\alpha+\sigma+\tau) - 2\sqrt{1+6(\alpha+\sigma)}(1+3\tau) + 2 \leq 0.$$

The left-hand side is minimized, for

$$\tau = \frac{1}{3}\left(\sqrt{1+6(\alpha+\sigma)} - 1\right).$$

Thus, for this choice of τ condition 2 is satisfied so we can successfully apply the LLL-algorithm.

From the LLL-reduced basis, we construct two polynomials $f_1(x,y)$, $f_2(x,y)$ with the common root (x_0, y_0) over the integers. By the heuristic assumption, the resultant $res_x(f_1, f_2)$ is not zero and we can find $y_0 = p + q - 1$ using standard root finding algorithms. This gives us the factorization of N.

To conclude the proof, we need to show that every step of the method can be done in time polynomial in log(N). The LLL-algorithm runs in polynomial time, since the basis matrix B has constant dimension (fixed by m) and its entries are bounded by a polynomial in N. Additionally, $res_x(f_1, f_2)$ has constant degree and coefficients bounded by a polynomial in N. Thus, every step can be done in polynomial time. □

We would like to make two considerations. The first is that when $\sigma = 0$, we get the same result of (Blömer and May, 2003). Indeed, our method is a generalization of it. The second is that we obtain the same bound of (Joye and Lepoint, 2012), but our approach is more effective in practice. As we will show in Section 6.1, we are able to get closer to the theoretical bound by using smaller lattices.

4.2 Partial Information on MSB of d^*

In this section, we prove that if the attacker knows a sufficiently large number of most significant bits of the protected exponent, then she can factor N.

To prove this result, we show how the partial knowledge on d^* can be used to construct an approximation of p that allows to apply Theorem 3.

The advantage of this approach compared to (Joye and Lepoint, 2012) is that it does not rely on the heuristic assumption 1 and yields to a better bound.

Theorem 5. *Let (N,e) be an RSA public key with $e = N^{\alpha}$ and $d^* = d + \ell\phi(N)$ for some $\ell = N^{\sigma}$ with $\sigma > 0$ and $N^{\alpha+\sigma} < 2N^{\frac{3}{8}}$. Suppose that $|p-q| \geq cN^{\frac{1}{2}}$, for some $c \leq \frac{1}{2}$, and suppose we are given an approximation $\widetilde{d^*}$ of d^* such that*

$$|d^* - \widetilde{d^*}| \leq cN^{\frac{1}{4}+\sigma}.$$

Then we can find the factorization of N in time polynomial in $\log N$.

Notice that, like in Theorem 4, we have $ed^* - 1 = k^*\phi(N)$ with $k^* = k + e\ell$.

In order to prove Theorem 5 we need first to prove the following lemma.

Lemma 1. *With $N^{\alpha+\sigma} < 2N^{\frac{3}{8}}$, given $\widetilde{d^*}$ such that $|d^* - \widetilde{d^*}| \leq \frac{1}{4}N^{1-\alpha}$ then the approximation $\widetilde{k^*} := \left\lceil \frac{e\widetilde{d^*}-1}{N+1} \right\rceil$ of k^* is exact.*

Proof. This proof follows the same strategy used in the proof of Theorem 6 of (Blömer and May, 2003). Note that

$$|k^* - \widetilde{k^*}| < \left| \frac{ed^*-1}{\phi(N)} - \frac{e\widetilde{d^*}-1}{N+1} \right|$$
$$< \left| \frac{(ed^*-1)(N+1) - (e\widetilde{d^*}-1)(N+1-(p+q))}{\phi(N)(N+1)} \right|.$$

Then, given that $\phi(N) > N/2$, $p+q \leq 3N^{\frac{1}{2}}$, $N^2 + N > N^2$ and $d^* < 2N^{1+\sigma}$, we obtain

$$\begin{aligned}
|k^* - \widetilde{k^*}| &< \left| \frac{e(d^* - \widetilde{d^*})}{\phi(N)} \right| + \left| \frac{(p+q)(e\widetilde{d^*}-1)}{\phi(N)(N+1)} \right| \\
&< \left| \frac{\frac{1}{4}N^{\alpha}N^{1-\alpha}}{\frac{N}{2}} \right| + \left| \frac{6N^{\frac{1}{2}+\alpha+1+\sigma}}{\frac{N}{2}(N+1)} \right| \\
&< \frac{1}{2} + 12N^{-\frac{1}{2}+\frac{3}{8}} < \frac{1}{2} + \frac{12}{N^{\frac{1}{8}}}
\end{aligned}$$

With RSA parameters, we have $12 \ll N^{1/8}$, so we can safely assume $|k^* - \widetilde{k^*}| < 1$. But the difference between two integers is an integer, thus we can conclude that it is zero, therefore $\widetilde{k^*} = k^*$. □

It is worth to observe two facts: first, the bound $|d^* - \widetilde{d^*}| \leq \frac{1}{4}N^{1-\alpha}$ requires the attacker to get the $(\log_2(N^{\sigma+\alpha}) + 2)$ most significant bits of d^*, a result which holds even for $\sigma = 0$ (i.e. $d^* = d$); second, the assumption $N^{\alpha+\sigma} < 2N^{\frac{3}{8}}$ of Lemma 1 always holds for our choice of RSA parameters.

We can now prove Theorem 5.

Proof of theorem 5. We begin by applying Lemma 1 to obtain the value of k^*. The condition $|d^* - \widetilde{d^*}| \leq \frac{1}{4}N^{1-\alpha}$ of the lemma is always satisfied by our choices of RSA parameters because $\frac{1}{2}N^{\frac{1}{4}+\sigma} \ll \frac{1}{4}N^{1-\alpha}$, since $N^{\sigma} < 2N^{\frac{1}{8}}$ and $N^{\alpha} < 2N^{\frac{1}{4}}$.

We can define an approximation $\tilde{s}$ of $s = p+q$ as

$$\tilde{s} := 1 + N - \frac{e\widetilde{d^*}-1}{k^*}$$

Reminding that k^*, with the assumption of $\sigma > 0$, is lower bounded by $N^{\alpha+\sigma}$, we obtain

$$|s - \tilde{s}| = \left| \frac{e}{k^*}\left(d^* - \widetilde{d^*}\right) \right| \leq \frac{N^{\alpha}}{N^{\alpha+\sigma}} cN^{\frac{1}{4}+\sigma} \leq cN^{\frac{1}{4}}.$$

We use $\tilde{s}$ to define

$$\tilde{p} := \frac{1}{2}\left(\tilde{s} + \sqrt{\tilde{s}^2 - 4N}\right)$$

as an approximation of p.

Without loss of generality, following Appendix B of (Boneh et al., 1998), we now assume that $\tilde{s} \geq s$, so that $\tilde{p} \geq p$.

Observe that

$$\begin{aligned}
\tilde{p} - p &= \frac{1}{2}(\tilde{s} - s) + \frac{1}{2}\left(\sqrt{\tilde{s}^2 - 4N} - \sqrt{s^2 - 4N}\right) \\
&= \frac{1}{2}(\tilde{s} - s) + \frac{(\tilde{s}+s)(\tilde{s}-s)}{2\left(\sqrt{\tilde{s}^2 - 4N} + \sqrt{s^2 - 4N}\right)}
\end{aligned}$$

Since $\tilde{s} \geq s$, we have $\tilde{s}^2 - 4N \geq s^2 - 4N = (p-q)^2$ and $|p-q| \geq cN^{\frac{1}{2}}$ with $c \leq \frac{1}{2}$.

Noting that $\tilde{s} \leq s + cN^{\frac{1}{4}}$, we have

$$\tilde{s} + s \leq 2s + cN^{\frac{1}{4}} \leq 2(p+q) + N^{\frac{1}{4}} \leq 6N^{\frac{1}{2}} + N^{\frac{1}{4}} \leq 7N^{\frac{1}{2}}.$$

It follows that

$$\begin{aligned}
\tilde{p} - p &\leq \frac{1}{2}(\tilde{s} - s) + \frac{(\tilde{s}+s)(\tilde{s}-s)}{4(p-q)} \\
&\leq \frac{1}{2}cN^{\frac{1}{4}} + \frac{(7N^{\frac{1}{2}})(cN^{\frac{1}{4}})}{4cN^{\frac{1}{2}}} \\
&\leq \frac{1}{4}N^{\frac{1}{4}} + \frac{7}{4}N^{\frac{1}{4}} \\
&\leq 2N^{\frac{1}{4}}
\end{aligned}$$

Since the approximation $\tilde{p}$ satisfies the hypothesis of Theorem 3 with $k = 1$, we can find the factorization of N in time polynomial in $\log N$. □

From Theorem 5 we can recover the minimum number of known MSB bits required. In accordance to previous sections we define this quantity as $\log_2 M$ where M is defined as

$$M = \frac{d^*}{|d^* - \widetilde{d}^*|} = \frac{2N^{1+\sigma}}{\frac{1}{2}N^{\frac{1}{4}+\sigma}} = 4N^{\frac{3}{4}}. \quad (3)$$

It is important to underline that this bound is not affected by the size of α and σ as long as the condition of Lemma 1 holds. In fact, while it might seem counter-intuitive, the presence of the countermeasure (i.e. $\sigma > 0$) improves the theoretical bound $|d - \tilde{d}| \leq cN^{\frac{1}{4}-\alpha}$ of Theorem 3.3 of (Boneh et al., 1998). However, this difference was not shown in the experimental results, probably due to low value of α when $e = 2^{16} + 1$.

Also note that Theorem 5 provides a significant improvement over the bound of (Joye and Lepoint, 2012). In fact, for $\alpha + \sigma \leq \frac{1}{2}$ (which is always true in our setting), their bound is $|d^* - \widetilde{d}^*| \leq N^{\alpha+\sigma}$, which would require knowledge of $\log_2(N^{1-\alpha})$ bits.

Attack using both MSB and LSB of $\mathbf{d}^*$ We want to briefly analyze also the case where the attacker might be able to detect bits in different positions of d^*. In this scenario, the attacker could obtain enough most significant bits to satisfy Lemma 1 and obtain $\frac{1}{4}\log_2 N$ least significant bits to recover half of the bits of p and factor N, as shown in (Boneh et al., 1998). Thus, the knowledge of only $(\log_2(N^{\frac{1}{4}+\sigma+\alpha}) + 2 + \varepsilon)$ bits and the resolution of an univariate equation are required. We don't describe the attack in details because, once k^* is recovered applying Lemma 1, it reduces to the method of (Boneh et al., 1998). Thus, we remind the reader to it. In Section 6, we will provide experimental results.

5 ATTACKS ON CRT-RSA

In this section we present two attacks on CRT-RSA implementations, where we target exponentiation by d_p^*. One is based on the knowledge of the most significant bits of the CRT private exponent and one is based on the knowledge of its least significant bits. We assume that the private exponent d_p is full-size (with respect to p) and that it is masked by a random multiple ℓ of $(p-1)$, for some $\ell \geq 0$. When $\ell = 0$ clearly $d_p^* = d_p$, that means that no countermeasure is applied.

5.1 Partial Information on LSB of $\mathbf{d_p^*}$

In this section, we assume that the attacker is able to recover the least significant bits of the secret d_p^*. We can write $d_p^* = d_1 \cdot M + d_0$ where d_0 is known to the attacker while d_1 is unknown. The integer M is a power of two and represents the bound on the known part.

We prove that if the attacker knows a sufficiently large number of least significant bits, then she can factor N.

To prove our result we use a method presented by Herrmann and May to find the solutions of a bivariate linear equation modulo p (Herrmann and May, 2008).

Theorem 6. *Let (N,e) be an RSA public key with $e = N^{\alpha}$. Let $d_p = d \bmod p-1$ and let $d_p^* = d + \ell(p-1)$ for some $\ell = N^{\sigma}$ with $\sigma \geq 0$. Suppose that $N^{\alpha+\sigma} \leq N^{\frac{1}{\sqrt{2}}-\frac{1}{2}}$ and that we are given d_0 and M satisfying $d_0 = d_p^* \bmod M$ with*

$$M \geq N^{1-\frac{1}{\sqrt{2}}+\alpha+2\sigma+\varepsilon},$$

for some $\varepsilon > 0$. Then, under Assumption 1, we can find the factorization of N (in time polynomial in $\log N$).

Proof. We start from the equation

$$ed_p - 1 = k_p(p-1).$$

Since $d_p^* = d_p + \ell(p-1)$, we obtain

$$ed_p^* - 1 = (k_p + e\ell)(p-1).$$

Let k_p^* denote $k_p + e\ell$. By writing $d_p^* = d_1 M + d_0$, we obtain the following equation

$$eMd_1 + k_p^* + ed_0 - 1 = k_p^* p.$$

It follows that the bivariate polynomial

$$f_p(x,y) = eMx + y + ed_0 - 1$$

has root $(x_0, y_0) = (d_1, k_p^*)$ modulo p.
In order to bound y_0, notice that

$$k_p^* = \frac{ed_p^* - 1}{(p-1)} < e\left(\frac{d_p + \ell(p-1)}{(p-1)}\right) < e(1+\ell) \leq 2N^{\alpha+\sigma}.$$

Additionally, recall that $d_1 = \frac{d_p^*}{M} - d_0$.
We can set bounds $X = N^{\frac{1}{\sqrt{2}}-\frac{1}{2}-\alpha-\sigma}$ and $Y = 2N^{\alpha+\sigma}$ so that $x_0 \leq X$ and $y_0 \leq Y$.
To construct the lattice, we consider the following auxiliary polynomials:

$$\overline{f} = x + Ry + R(ed_0 - 1) \text{ where } R = (eM)^{-1} \bmod N$$

$$g_{k,i} = y^i \overline{f}^k N^{\max\{t-k,0\}},\ k = 0,\ldots,m; i = 0,\ldots,m-k$$

for some integers m and t, where $t = \tau m$ has to be optimized.

All integer linear combinations of these polynomials share the root (x_0, y_0) modulo p^t. Thus, the first condition of Theorem 2 is satisfied. In order to satisfy the second condition we have to find a short vector in the lattice L, spanned by $g_{k,i}(xX, yY)$. In particular, this vector shall have a norm smaller than $\frac{p^t}{\sqrt{\dim L}}$.

The second condition of Theorem 2 is satisfied when equation (1) holds, i.e. when

$$\det L \leq N^{\frac{1}{2}\tau m(n-1)}. \tag{4}$$

A straightforward computation shows that $n = \frac{1}{2}(m^2+3m+2)$ and that

$$\det L(M) = (XY)^{\frac{1}{6}(m^3+3m^2+2m)} N^{\frac{1}{6}m\tau(m\tau+1)(4+3m-m\tau)}.$$

Thus, condition (4) becomes

$$(XY)^{\frac{1}{6}(m^3+3m^2+2m)} \leq N^{\frac{1}{4}\tau m(m^2+3m)-\frac{1}{6}m\tau(m\tau+1)(4+3m-m\tau)}$$

that reduces to

$$XY \leq N^{\frac{1}{2}(3\tau+2\tau^3-6\tau^2)}.$$

Since $XY = 2N^{\frac{1}{\sqrt{2}}-\frac{1}{2}}$, the above condition is satisfied if

$$\frac{1}{\sqrt{2}} - \frac{1}{2} - \frac{1}{2}(3\tau+2\tau^3-6\tau^2) \leq 0.$$

The left-hand side is minimized for $\tau = 1 - \frac{1}{\sqrt{2}}$. For this choice of τ condition (4) is satisfied, so we can successfully apply LLL-algorithm and then find the root (d_1, k_p^*). From this values, we can obtain $p-1$ and then the factorization of N.

To conclude the proof, we need to show that every step of the method can be done in time polynomial in log(N). The LLL-algorithm is polynomial in the dimension of the matrix, that is $O(m^2)$, and in the bit-size of its entries, that are $O(m \log N)$. Additionally, $res_y(f_1, f_2)$ has constant degree and coefficients bounded by a polynomial in N. Thus, every step can be done in polynomial time. □

5.2 Partial Information on MSB of $\mathbf{d_p^*}$

In this section, we prove that if the attacker knows a sufficiently large number of most significant bits of the protected exponent d_p^*, then she can factor N.

To prove this result, we show how the partial knowledge on d_p^* can be used to construct an approximation of a multiple of p that allows to apply Theorem 3.

Theorem 7. *Let (N,e) be an RSA public key with $e = N^\alpha$. Let $d_p = d \bmod p-1$ and let $d_p^* = d_p + \ell(p-1)$, for some $\ell = N^\sigma$ with $\sigma \geq 0$. Suppose that $N^{\alpha+\sigma} \leq \frac{1}{2}N^{\frac{1}{4}}$ and that we are given an approximation $\widetilde{d_p^*}$ of d_p^* such that*

$$|d_p^* - \widetilde{d_p^*}| \leq N^{\frac{1}{4}-\alpha}.$$

Then, we can find the factorization of N in time polynomial in $\log N$.

Proof. We start from equation

$$ed_p^* - 1 = k_p^*(p-1)$$

with $k_p^* = k_p + \ell e$.

Note that $k_p^* \leq 2N^{\alpha+\sigma} < \frac{1}{2}N^{\frac{1}{2}}$ implies that q can't divide k_p^*.

We compute an approximation

$$\widetilde{k_p^* p} := e\widetilde{d_p^*} - 1$$

of $k_p^* p$, up to an additive error of at most

$$\begin{aligned} |k_p^* p - \widetilde{k_p^* p}| &= |ed_p^* - 1 + k_p^* - e\widetilde{d_p^*} + 1| \\ &= |e(d_p^* - \widetilde{d_p^*}) + k| \leq N^{\frac{1}{4}} + 2N^{\alpha+\sigma} \leq 2N^{\frac{1}{4}}. \end{aligned}$$

Since the approximation $\widetilde{k_p^* p}$ satisfies the hypothesis of Theorem 3, we can find the factorization of N in time polynomial in $\log N$. □

The bound of Theorem 7 implies that an attacker has to know at least $\log_2 M$ bits, where

$$M = \frac{d_p^*}{|d_p^* - \widetilde{d_p^*}|} = \frac{2N^{1+\sigma}}{N^{\frac{1}{4}-\alpha}} = 2N^{\frac{3}{4}+\alpha+\sigma} \tag{5}$$

This bound holds when the condition $N^{\alpha+\sigma} \leq \frac{1}{2}N^{\frac{1}{4}}$ holds, which is not always the case in our settings. For example an RSA modulus of 1024 bit with $\log_2 e = 256$ and $log_2 \ell = 128$ will have $N^{\alpha+\sigma} \leq 2N^{\frac{3}{8}}$. For these cases we are unaware of successful applications of Coppersmith's method.

In (Lu et al., 2014) Section 4 it is presented a novel technique for the CRT case with better bound but with the requirement to have d_p not full size. This requirement also implies that no countermeasure is applied.

6 EXPERIMENTAL RESULTS

Here we present experimental results for the attacks described in previous sections.

We consider RSA applications with 2048-bit modulus N and public exponent $e = 2^{16}+1$, since this is the most common choice made for real implementations.

In addition we assume that a random multiple ℓ of $\phi(N)$ (or of $(p-1)$ for CRT-RSA applications) is added to the private exponent d (respectively d_p).

For each dimension of ℓ, we first report the theoretical bound on the minimum number of bits of the secret key that the attacker needs to know to recover it entirely. This values are derived from theorems we have proved in previous sections.

Then, we report the average minimum number of bits that we really needed in our tests. In fact, theoretical bounds are reached when the lattice dimension goes to infinity. In general, the smaller is the number of known bits, the bigger the lattice shall be. To concretely mount an attack, one needs to construct a lattice whose dimension is such that the LLL-algorithm runs in practical time. Recall that the running time of LLL-algorithm depends on the lattice dimension and on the dimension of the entries of its matrix-basis. Since the dimension of the entries depends on the bounds X_i and on the modular polynomial used, the LLL-algorithm may have different running times for the same lattice dimension.

We decided to fix an upper bound on the dimension of the lattices we constructed. We chose the threshold 80 as a tradeoff between efficiency and effectiveness of our attacks. Indeed, this choice allows us to get closer to the theoretical bounds as opposed to smaller dimensions. On the other hand, 80 is small enough to make the LLL-algorithm running in practical time.

We fixed the same threshold for all attacks in order to compare their effectiveness when using the same lattice dimension.

We implemented our methods with the SAGE computer-algebra system (Stein et al., 2014) and run it on a 3GHz Intel Core i5.

With the exception of the CRT-MSB case, where we used only 10 experiments, for all other attacks we ran 100 experiments generating different key pairs and different values of ℓ. We report the average values obtained from these experiments.

6.1 Results with known LSB of d*

In Section 4.1 we proved that if the attacker knows a sufficiently large number of least significant bits of d^*, then she can factor N.

For generating the lattices, we used $m = 11$ and $t = \tau m$, where τ is defined in the proof of Theorem 4. Notice that τ is always very small resulting in $t = 0$ for each experiment. Thus, the dimension of the lattice is fixed and equal to 78.

In Table 1 we present our results. For different dimensions of ℓ we report theoretical and experimental bounds on the minimum number of leaked bits necessary to mount the attack. Then we report the lattice dimension that allowed us to get the corresponding experimental bound and finally the running time of LLL-algorithm for these lattices.

The difference between theoretical and experimental bounds is of very few bits and the LLL-algorithm's running time is really small.

Table 1: Experimental results for partial key exposure attack given least significant bits of the secret exponent $d^* = d + \ell\phi(N)$. The modulus N is 2048-bit long and $e = 2^{16} + 1$.

$\log_2 \ell$	*theo. bound*	*exp. bound*	dim(L)	LLL
0	1040	1043	78	18 s
10	1060	1063	78	19 s
32	1103	1106	78	22 s
64	1164	1171	78	50 s
100	1232	1243	78	70 s

It is worth to say that for $\ell = 0$ and small e, the attack in (Boneh et al., 1998) is more effective than our attack. Indeed the $n/4$ least significant bits of d are sufficient to factor N. However their attack requires a brute-force search on k, that is allowed only when $e + e\ell$ is small. Thus, for $e = 2^{16} + 1$ and $\ell = 0$ their method is more effective than ours. But, with the introduction of exponent blinding, or for larger dimension of e, their method can't be applied, because the brute force-search becomes impractical.

Now, we compare our approach and the approach of (Joye and Lepoint, 2012) for the same scenario.

We use a bivariate polynomial instead of a trivariate polynomial, thus we perform a single resultant computation, instead of three.

In order to compare the two approaches, we report experimental results obtained using the same parameter choices made by the authors in (Joye and Lepoint, 2012). Specifically, we consider 1000-bit modulus N, public exponent $e = 2^{16} + 1$ and $\ell \in \{10, 100, 200, 300\}$.

As shown in Table 2, the theoretical bound is the same, but our approach allows us to get closer to it. Moreover, we do it by using smaller lattices.

6.2 Results with known MSB of d*

In this section we report experimental results on factoring with knowledge of the most significant bits of the protected private exponent.

Since this method uses an univariate polynomial, it is possible, in theory, to match the theoretical limit, although the lattice dimension would make LLL highly impractical. By imposing the threshold for the maximum dimension of the lattice equal to 80, the LLL-algorithm's running time is about 2 hours. For constructing such a lattice, we used $m = 40$ and $t = 40$.

In Table 3 we present our results. We report theoretical and experimental bounds on the number of

Table 2: Comparison between the approach of (Joye and Lepoint, 2012) and our approach for partial key exposure attack given least significant bits of the secret exponent $d^* = d + \ell\phi(N)$. The modulus N is 1000-bit long and $e = 2^{16}+1$.

$\log_2 \ell$	Approach of (Joye and Lepoint, 2012)				Our approach			
	theo. bound	*exp. bound*	dim(L)	LLL	*theo. bound*	*exp. bound*	dim(L)	LLL
10	535	580	16	1 sec	535	540	10	1 sec
100	700	760	16	1 sec	700	720	10	1 sec
200	871	960	16	1 sec	871	920	10	1 sec
300	1033	1160	16	1 sec	1033	1120	10	1 sec

Table 3: Experimental results for partial key exposure attack given most significant bits of the secret exponent $d^* = d + \ell\phi(N)$. The modulus N is 2048-bit long and $e = 2^{16}+1$.

$\log_2 \ell$	*theo. bound*	*exp. bound*	dim(L)	LLL
0	1555	1555	80	112 m
10	1538	1555	80	112 m
32	1538	1555	80	112 m
64	1538	1555	80	112 m
100	1538	1555	80	112 m

Table 4: Experimental results for partial key exposure attack given most and least significant bits of the secret $d^* = d + \ell\phi(N)$. The modulus N is 2048-bit long and $e = 2^{16}+1$.

$\log_2 \ell$	*theo. bound*	*exp. bound*	dim(L)	LLL
0	17+514	17+526	80	2h 27m
10	27+514	27+526	80	2h 27m
32	49+514	49+526	80	2h 27m
64	81+514	81+526	80	2h 27m
100	117+514	117+526	80	2h 27m

leaked bits, the dimension of the lattice and the running time of the LLL-algorithm.

The experiments confirmed the independence of the bound with respect to the dimension of the random integer ℓ.

Unfortunately, in this case we cannot compare our approach with the approach of (Joye and Lepoint, 2012), because they didn't provide any experimental result respecting our assumptions. In fact, they use very large values of ℓ, namely 500, 600 or 700-bit long for a modulus N of size 1000 bits. These unrealistic settings do not satisfy our requirement of Lemma 1 for $N^{\alpha+\sigma} \leq 2N^{\frac{3}{8}}$. In any case, our approach improves their bound, as said in section 4.2.

Results using both MSB and LSB. As said in section 4.2, it is possible to mount an attack knowing both MSB and LSB of d^*. An univariate polynomial is constructed and its root is found by constructing a lattice as in the proof of Theorem 3. In Table 4 we provide some experimental results for this method.

6.3 Results with known LSB of $\mathbf{d_p^*}$

In this section we report experimental results for CRT-RSA applications when the attacker knows the least significant bits of the blinded private exponent $d_p^* = d_p + \ell(p-1)$.

To get close to the theoretical bound, the lattice dimension has to be significantly increased. But this makes the LLL-algorithm's running time highly impractical. By setting the threshold 80 for the lattice dimension, the LLL-algorithm's running time is about 13 minutes.

In Table 5 we present our results. We report theoretical and experimental bounds on the number of leaked bits, the dimension of the lattice and the running time of the LLL-algorithm.

Table 5: Experimental results for partial key exposure attack against CRT-RSA, given least significant bits of the secret exponent $d_p^* = d_p + \ell(p-1)$. The modulus N is 2048-bit long and $e = 2^{16}+1$.

$\log_2 \ell$	*theo. bound*	*exp. bound*	dim(L)	LLL
0	617	691	78	8 m
10	637	712	78	10 m
32	681	758	78	13 m
64	745	822	78	17 m
100	817	894	78	18 m

In this case, the difference between theoretical and experimental bounds is about 80 bits. Given a smaller number of leaked bits one can still mount the attack by constructing bigger lattices, but the computation will need more time to end. For example, by setting $t = 5$ and $m = 18$ it is sufficient to obtain 50 bits more than the theoretical bound to solve. But the corresponding lattice dimension is 190, which makes the LLL-algorithm end in about one day. By setting $t = 7$ and

$m = 24$ it is sufficient to obtain 40 bits more than the theoretical bound to solve. But the lattice dimension is around 500 and we think that the LLL-algorithm would be highly impractical in this case.

Notice that for $\ell = 0$ and small e, Blömer and May show that a quarter of d_p is sufficient to the attacker to factor N (Blömer and May, 2003). To prove their result, they use a brute-force search on k_p, that is allowed only when $e + e\ell$ is small. Thus, for $e = 2^{16} + 1$ and $\ell = 0$ their method is better than our method, since a smaller number of leaked bits are sufficient to factor N. But, for larger dimension of e and when $\ell > 0$ their method is no more effective because the brute force-search becomes unfeasible.

6.4 Results with known MSB of $\mathbf{d_p^*}$

Here we report experimental results for CRT-RSA applications when the attacker knows the most significant bits of the protected private exponent.

Also in this case we imposed the threshold 80 for the lattice dimension, which allowed us to run the LLL-algorithm in practical time. We constructed lattices by using $m = 40$ and $t = 40$.

In Table 6 we report the theoretical and experimental number of leaked bits, the lattice dimension and the running time of LLL-algorithm.

Table 6: Experimental results for partial key exposure attack given most significant bits of the CRT secret exponent $d_p^* = d + \ell(p-1)$.

$\log_2 \ell$	*theo. bound*	*exp. bound*	dim(L)	LLL
0	528	540	80	3h 03m
10	537	550	80	3h 59m
32	560	573	80	4h 23m
64	591	604	80	4h 52m
100	628	640	80	6h 13m

As opposite to the case based on LSB, this method is the most effective also for $\ell = 0$. Indeed, our method is a generalization of (Blömer and May, 2003), thus for $\ell = 0$ we obtain their original result which is the most effective method in literature for this scenario.

7 CONCLUSIONS

We presented some methods to mount partial key exposure attacks on RSA with exponent blinding. We investigated both RSA and CRT-RSA, focusing on practical settings for the exponents and the blinding factor ℓ. In particular, we focused on public exponent e such that $3 \leq e < 2^{256}$, combining the upper bound provided by NIST with the frequent value of 3. Additionally, we focused on full size private exponents and $\ell < 2^{128}$, as commonly used in real implementations.

We derived sufficient conditions to successfully mount partial key exposure attacks in different scenarios and validated them providing numerical experiments, using N of size 2048 and $e = 2^{16} + 1$, which is the most commonly used setting in real implementations.

As for RSA, we improved the results of (Joye and Lepoint, 2012) with the aim of reducing the number of bits to be recovered by the adversary through side-channel. In particular, when least significant bits are exposed, our approach allows to get closer to the theoretical bound by using smaller lattices, as shown in Table 2. Whereas, when most significant bits are exposed, we presented a method that does not rely on the heuristic assumption and that provides better bounds, as shown in Section 4.2.

Additionally, we provided novel results for the particular case where the adversary is able to recover non-consecutive portions of the private information.

As for CRT-RSA with exponent blinding, we provided novel results for both scenarios when either least or most significant bits are exposed.

In Table 7, we recap the numerical results we obtained from our experiments. For each dimension of ℓ we provide the minimum number of bits of the protected exponent that is sufficient to the attacker to successfully break the system.

With the only exception of the RSA attack based on most significant bits, the number of known bits depends on the bit-size of the blinding factor ℓ.

ACKNOWLEDGEMENTS

This work was partly supported by the Italian MIUR project SecurityHorizons (c.n. 2010XSEMLC).

REFERENCES

Blömer, J. and May, A. (2003). New partial key exposure attacks on RSA. In Boneh, D., editor, *Advances in Cryptology - CRYPTO 2003, Proceedings*, volume 2729 of *LNCS*, pages 27–43. Springer.

Boneh, D., Durfee, G., and Frankel, Y. (1998). An attack on RSA given a small fraction of the private key bits. In Ohta, K. and Pei, D., editors, *Advances in Cryptology - ASIACRYPT 1998, Proceedings*, volume 1514 of *LNCS*, pages 25–34. Springer.

Table 7: The number of bits that the attacker needs to know to successfully mount partial key exposure attacks. The modulus N is 2048-bit long and the public exponent is $e = 2^{16} + 1$. The private exponent is blinded using the random factor ℓ.

$\log_2 \ell$	LSB	MSB	MSB+LSB	CRT-LSB	CRT-MSB
0	1043	1555	17+526	691	540
10	1063	1555	27+526	712	550
32	1106	1555	49+526	758	573
64	1171	1555	81+526	822	604
100	1243	1555	117+526	894	640

Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., and Verneuil, V. (2010). Horizontal correlation analysis on exponentiation. In Soriano, M., Qing, S., and López, J., editors, *Information and Communications Security - ICICS 2010, Proceedings*, volume 6476 of *LNCS*, pages 46–61. Springer.

Coppersmith, D. (1996a). Finding a small root of a bivariate integer equation; factoring with high bits known. In (Maurer, 1996), pages 178–189.

Coppersmith, D. (1996b). Finding a small root of a univariate modular equation. In (Maurer, 1996), pages 155–165.

Coron, J. (1999). Resistance against differential power analysis for elliptic curve cryptosystems. In Koç, Ç. K. and Paar, C., editors, *Cryptographic Hardware and Embedded Systems - CHES 1999, Proceedings*, volume 1717 of *LNCS*, pages 292–302. Springer.

Ernst, M., Jochemsz, E., May, A., and de Weger, B. (2005). Partial key exposure attacks on RSA up to full size exponents. In Cramer, R., editor, *Advances in Cryptology - EUROCRYPT 2005, Proceedings*, volume 3494 of *LNCS*, pages 371–386. Springer.

Fouque, P., Kunz-Jacques, S., Martinet, G., Muller, F., and Valette, F. (2006). Power attack on small RSA public exponent. In Goubin, L. and Matsui, M., editors, *Cryptographic Hardware and Embedded Systems - CHES 2006, Proceedings*, volume 4249 of *LNCS*, pages 339–353. Springer.

Herrmann, M. and May, A. (2008). Solving linear equations modulo divisors: On factoring given any bits. In Pieprzyk, J., editor, *Advances in Cryptology - ASIACRYPT 2008, Proceedings*, volume 5350 of *LNCS*, pages 406–424. Springer.

Howgrave-Graham, N. (1997). Finding small roots of univariate modular equations revisited. In Darnell, M., editor, *Cryptography and Coding, 6th IMA International Conference 1997, Proceedings*, volume 1355 of *LNCS*, pages 131–142. Springer.

Joye, M. and Lepoint, T. (2012). Partial key exposure on RSA with private exponents larger than N. In Ryan, M. D., Smyth, B., and Wang, G., editors, *Information Security Practice and Experience - ISPEC 2012, Proceedings*, volume 7232 of *LNCS*, pages 369–380. Springer.

Kerry, C. F., Secretary, A., and Director, C. R. (2013). FIPS PUB 186-4 FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Digital Signature Standard (DSS).

Kocher, P. C. (1996). Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Koblitz, N., editor, *Advances in Cryptology - CRYPTO 1996, Proceedings*, volume 1109 of *LNCS*, pages 104–113. Springer.

Kocher, P. C., Jaffe, J., and Jun, B. (1999). Differential power analysis. In Wiener, M. J., editor, *Advances in Cryptology - CRYPTO 1999, Proceedings*, volume 1666 of *LNCS*, pages 388–397. Springer.

Lenstra, A., Lenstra, H., and Lovász, L. (1982). Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534.

Lu, Y., Zhang, R., and Lin, D. (2014). New partial key exposure attacks on CRT-RSA with large public exponents. In Boureanu, I., Owesarski, P., and Vaudenay, S., editors, *Applied Cryptography and Network Security - ACNS 2014, Proceedings*, volume 8479 of *LNCS*, pages 151–162. Springer.

Maurer, U. M., editor (1996). *Advances in Cryptology - EUROCRYPT 1996, Proceeding*, volume 1070 of *LNCS*. Springer.

May, A. (2003). *New RSA vulnerabilities using lattice reduction methods*. PhD thesis, University of Paderborn.

Quisquater, J.-J. and Couvreur, C. (1982). Fast decipherment algorithm for rsa public-key cryptosystem. *Electronic Letters*, 18:905–907.

Stein, W. et al. (2014). *Sage Mathematics Software (Version 6.2)*. The Sage Development Team. http://www.sagemath.org.

Walter, C. D. (2001). Sliding windows succumbs to big mac attack. In Koç, Ç. K., Naccache, D., and Paar, C., editors, *Cryptographic Hardware and Embedded Systems - CHES 2001, Proceedings*, volume 2162 of *LNCS*, pages 286–299. Springer.

Wiener, M. J. (1990). Cryptanalysis of short rsa secret exponents. *IEEE Transactions on Information Theory*, 36:553–558.