Published in M.D. Ryan, B. Smyth, and G. Wang, Eds, *Information Security Practice and Experience (ISPEC 2012)*, vol. 7232 of *Lecture Notes in Computer Science*, pp. 369-380, Springer, 2012

Partial Key Exposure on RSA with Private Exponents Larger than N

Marc Joye[1] and Tancrède Lepoint[2,3,⋆]

[1] Technicolor, Security & Content Protection Labs
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France
marc.joye@technicolor.com
[2] CryptoExperts
41 boulevard des Capucines, 75002, Paris, France
[3] Laboratoire d'Informatique de l'École Normale Supérieure
45 rue d'Ulm, 75230 Paris Cedex 05, France
tancrede.lepoint@cryptoexperts.com

Abstract. In 1998, Boneh, Durfee and Frankel described several attacks against RSA enabling an attacker given a fraction of the bits of the private exponent d to recover all of d. These attacks were later improved and extended in various ways. They however always consider that the private exponent d is smaller than the RSA modulus N. When it comes to implementation, d can be enlarged to a value larger than N so as to improve the performance (by lowering its Hamming weight) or to increase the security (by preventing certain side-channel attacks). This paper studies this extended setting and quantifies the number of bits of d required to mount practical partial key exposure attacks. Both the cases of known most significant bits (MSBs) and least significant bits (LSBs) are analyzed. Our results are based on Coppersmith's heuristic methods and validated by practical experiments run through the SAGE computer-algebra system.

Keywords: RSA cryptosystem, cryptanalysis, key exposure, Coppersmith's methods, lattice reduction.

1 Introduction

For efficiency reasons, it might be tempting to select a small RSA private exponent d. In 1990, Wiener [30] showed that this results in an insecure system when $d < N^{0.25}$, where N denotes the RSA modulus. His attack made use of the continued fractions method. The bound was subsequently improved to $d < N^{0.292}$ by Boneh and Durfee [3] (see also [14, Section 3]) from powerful LLL-based techniques due to Coppersmith [8].

A related problem is that of partial key exposure: What is the fraction of the private exponent d that has to be made available to an attacker in order to break the system. This question was posed by Boneh, Durfee and Frankel [4]

⋆ This work was done while the author was with Technicolor.

in 1998. They present several attacks where the attacker gains knowledge of most significant bits of d or of least significant bits of d. (Observe that Wiener's attack corresponds to a partial key exposure where the most significant bits are known to be zero.) Further partial key exposure attacks are presented by Blömer and May in [2] for larger values of public exponent e. The attacks in [4] require $e < N^{1/2}$. For attacks that work up to full-size exponents e, we refer to the paper of Ernst, Jochemsz, May and de Weger [12].

One may argue that partial key exposure attacks are unimportant. If an attacker is able to recover some bits of d then it should likewise be able to recover the entire private key d. While this may hold true in theory, in practice things are not so easy. Revealing some bits of d can be a lengthy and costly process. Partial key exposure attacks then facilitate the recovery of the entire private key. This especially applies to RSA implementations since it is likely that precautions were taken to prevent an adversary to obtain the private key d. Examples of implementation attacks include side-channel attacks [20, 18, 19]. Such attacks exploit differences in running times, power consumption traces, or other side channels resulting from the execution of the cryptographic algorithm. Yet another use case of partial key exposure is in covert communication channels (a.k.a. subliminal channels) [27, 28, 31]. A covert channel enables users to exchange secret information (e.g., an RSA private key) through messages that appear to be innocuous. Partial key exposure then reduces the number of exchanged messages.

All partial key exposure results on RSA presented so far have in common that the private exponent d is defined as an element in $\mathbb{Z}^*_{\phi(N)}$, where $\phi(N)$ denotes Euler's totient function of N —namely, the order of the multiplicative group of integers modulo N. In a number of implementations, a multiple of $\phi(N)$ is added to d prior to the exponentiation. In more detail, the RSA exponentiation $x^d \bmod N$ is carried out as $x^{d+k\phi(N)} \bmod N$ for some $k > 0$, that is, with an exponent whose size is at least that of N. There are mainly two reasons to do so. One of them is to offer better resistance against implementation attacks like side-channel attacks (e.g., [9, § 5.1]). The other reason is efficiency. An appropriate choice of k can lower the Hamming weight of the exponent and reduce the total number of multiplications, leading to an expected performance improvement of up to 9.3% [5].

This paper deals with private exponents d that are larger than N (or more exactly than $\phi(N)$). We follow the heuristic strategy put forward by Jochemsz and May [16, 17] for solving multivariate polynomials with small integer or modular roots. The main advantage resides in its generality. The method is nevertheless heuristic in the sense that it is not guaranteed to succeed. We therefore ran many experiments with different parameter sets and none of them failed. Two practical cases are considered. The first case assumes that the attacker knows the most significant bits (MSBs) of d. Informally, we then show that if the public/private exponents verify $(e, d) \sim (N^\alpha, N^\beta)$ with $\beta \geqslant 1$ the fraction of d that is sufficient to recover it entirely is given by

$$\begin{cases} \dfrac{1-\alpha}{\beta} & \text{when } 1 < \alpha+\beta \leqslant \frac{3}{2} \\ \dfrac{2\beta - \alpha + 2\sqrt{(\alpha+\beta)\left(\alpha+\beta-\frac{3}{2}\right)}}{3\beta} & \text{when } \frac{3}{2} \leqslant \alpha+\beta < 2 \end{cases}.$$

In some scenarios, the least significant bits (LSBs) can be easier to obtain; for example, when the underlying exponentiation algorithm processes the bits of d from the right to the left. Informally, with the above notation, we then show that the fraction of d that is sufficient to recover it entirely is given by

$$\frac{(6\beta - 5) + 2\sqrt{6\alpha + (6\beta - 5)}}{6\beta} \quad .$$

The rest of this paper is organized as follows. In the next section, we introduce some useful background on lattice basis reduction. We also sketch the strategy of Jochemsz and May. We apply it to the cases of known MSBs in Section 3 and known LSBs in Section 4. Section 5 provides various data obtained through numerical experiments. Finally, we conclude in Section 6 and discuss some open issues for further research.

2 LLL and Multivariate Polynomials

2.1 Lattices

A *lattice* is a discrete additive subgroup of $\mathbb{R}^n$. For any integer lattice $L \neq \{0\}$, there exist $w \leqslant n$ linearly independent vectors $\boldsymbol{b_1}, \ldots, \boldsymbol{b_w}$ over $\mathbb{R}$ such that $L = \boldsymbol{b_1}\mathbb{Z} \oplus \cdots \oplus \boldsymbol{b_w}\mathbb{Z}$. This set of vectors is called a *basis* of the lattice. A lattice can be so represented by its basis matrix B; i.e., the matrix of the $\boldsymbol{b_i}$'s in the canonical basis of $\mathbb{R}^n$. The *determinant* of a lattice is defined as $\det(L) = (\det(BB^t))^{\frac{1}{2}}$. When the lattice is full-rank ($w = n$), the formula simplifies to $\det(L) = |\det B|$. The determinant of a lattice is well-defined since it is independent of the choice of the basis: lattice bases are obtained one from the others through a unimodular transformation (i.e., a multiplication by a matrix with determinant ± 1).

Among all the bases of a lattice L, some are 'better' than others. The goal of lattice reduction is to shorten the basis vectors and thus, since the determinant is invariant, to make them more orthogonal. The LLL algorithm, named after Lenstra, Lenstra and Lovász [21], produces in polynomial time a set of reduced basis vectors. The following lemma, as presented in [22], gives bounds on LLL-reduced basis vectors.

Lemma 1 (LLL). *Let L be a lattice of dimension w. In polynomial time, the LLL algorithm outputs reduced basis vectors $\boldsymbol{v_i}$, $1 \leqslant i \leqslant w$, satisfying*

$$\|\boldsymbol{v_1}\| \leqslant \|\boldsymbol{v_2}\| \leqslant \cdots \leqslant \|\boldsymbol{v_i}\| \leqslant 2^{\frac{w(w-1)}{4(w+1-i)}} \det(L)^{\frac{1}{w+1-i}} \quad .$$

2.2 Strategy for finding small roots of multivariate polynomials

In [6–8], Coppersmith presents rigorous methods based on LLL to find small roots of univariate modular polynomials or of bivariate integer polynomials. The methods can be extended to polynomials in more variables, but only heuristically.

The methods for finding small roots of a polynomial f mainly depend on the shape of its Newton polytope (in others words, of the monomials that appear in f). In [16], Jochemsz and May presents a heuristic strategy that applies to any multivariate polynomial with either modular or integer roots, based on Howgrave-Graham [15] lemma for the univariate case, and the improvements of Coron [10, 11]. We briefly review hereafter their root-finding strategy.

For the sake of illustration, consider without loss of generality a trivariate polynomial $f(x, y, z)$ over the integers. Let (x_0, y_0, z_0) be a (small) root of f, and let (X, Y, Z) be a upper bound of this root; i.e., $|x_0| < X$, $|y_0| < Y$ and $|z_0| < Z$. Define $W = \|f(xX, yY, zZ)\|_\infty$ the maximal coefficient (in absolute value) of $f(xX, yY, zZ)$. A basis B of a lattice L is defined via the so-called *shift polynomials* $x^i y^j z^k f(x, y, z)$ (resp. $x^i y^j z^k$) for $\{i, j, k\}$ determined by a set S (resp. $M \setminus S$) which depends on the monomials of f. The set M then consists of all the monomials that appear in the shift polynomials $x^i y^j z^k f(x, y, z)$. LLL reduction algorithm is then performed on B in order to reduce the lattice L. From a result of [16], provided that

$$X^{s_x} Y^{s_y} Z^{s_z} < W^s \,, \quad \text{where} \begin{cases} s_x = \sum_{x^i y^j z^k \in M \setminus S} i + (m-1) \\ s_y = \sum_{x^i y^j z^k \in M \setminus S} j + (m-1) \\ s_z = \sum_{x^i y^j z^k \in M \setminus S} k + (m-1) \end{cases} \quad \text{and } s = |S| - 1 \,,$$

the first vectors, say f_0 and f_1, of the reduced basis produced by LLL provide two polynomials with root (x_0, y_0, z_0) over the integers. The common roots of $\{f, f_0, f_1\}$ are revealed under the assumption that two variables can be eliminated from the system $\{f = 0, f_0 = 0, f_1 = 0\}$. This can be done through the evaluation of resultants or of Gröbner bases.

Again, it is worth remembering that the strategy is heuristic. It is assumed that the aforementioned use of resultants produces a non-zero univariate polynomial, for which finding integer roots is easy. So, in our example, f_0 and f_1 need to be algebraically independent. More generally, the following assumption is supposed to hold true for n-variate polynomials, $n \geqslant 3$.

Assumption 1. *The resultant computations for the polynomials f_i yield non-zero polynomials.*

This heuristic assumption has proven to be useful in many attacks (e.g., [4, 2, 10, 12, 16, 11, 17, 25, 24]).

3 Key Recovery from Known MSBs

Most key recovery attacks on RSA cryptosystem use a similar technique. The goal is to derive, from an RSA equation, a multivariate polynomial in some of the unknowns of RSA, like p, q, d or $\phi(N)$.

Let $N = pq$ be an RSA modulus where p and q are two equal-size (balanced) primes. The public exponent is denoted e and the corresponding private exponent

is $d = e^{-1} \bmod \phi(N)$. In this section, we assume that the attacker succeeded in getting the most significant bits of $d^* = d + \ell\phi(N)$ for some unknown integer $\ell > 0$. We write

$$d^* = \tilde{d} + d_0$$

where $\tilde{d}$ is a known approximation for d^* and d_0 is the value to be found. The upper bound on d_0 is defined as $d_0 \leqslant N^\delta$. The next theorem states how large δ can be in order to recover d_0 (and thus the entire private key d^*). Note that the knowledge of d^* yields a non-zero multiple of $\phi(N)$ as $ed^* - 1$ and consequently the two secret factors of N [23].

Theorem 1. *With the previous notations, suppose that $e \sim N^\alpha$ and $d^* \sim N^\beta$. Then under Assumption 1 and up to a small error factor ϵ, there exists, for sufficiently large N, a polynomial-time algorithm that computes all of d^*, provided that*

$$\delta \leqslant \begin{cases} \alpha + \beta - 1 - \epsilon & \text{for } 1 < \alpha + \beta < \frac{3}{2} \\ \frac{\alpha+\beta-\sqrt{4(\alpha+\beta)^2-6(\alpha+\beta)}}{3} - \epsilon & \text{for } \frac{3}{2} \leqslant \alpha + \beta < 2 \end{cases} .$$

The rest of this section will be devoted to the proof of Theorem 1. Notice also that it leads to the following result:

Corollary 1. *Using the notation of Theorem 1, recovering a fraction of d^* larger than*

$$\begin{cases} \frac{1-\alpha}{\beta} & \text{when } 1 < \alpha + \beta \leqslant \frac{3}{2} \\ \frac{2\beta-\alpha+2\sqrt{(\alpha+\beta)\left(\alpha+\beta-\frac{3}{2}\right)}}{3\beta} & \text{when } \frac{3}{2} \leqslant \alpha + \beta < 2 \end{cases} .$$

is sufficient to recover the entire private exponent d^* in polynomial time.

A graphical interpretation of this corollary, representing the fraction of d^* required to recover the entire exponent d^* in polynomial time, is depicted in Fig. 1.

Proof (of Corollary 1). This is a immediate consequence of Theorem 1 since the fraction of d^* corresponding to the unknown d_0 is given by $1 - \frac{\delta}{\beta}$. □

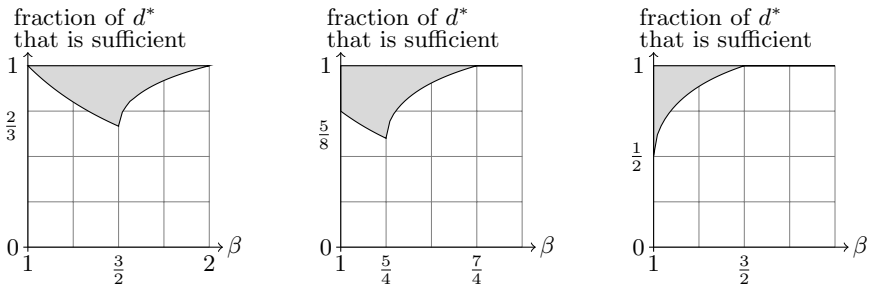
3.1 Preliminaries

Since $ed \equiv 1 \pmod{\phi(N)}$ and $d^* = d + \ell\phi(N)$, we can write $ed^* = 1 + k^*\phi(N)$ for some integer k^*, or equivalently,

$$e\tilde{d} + ed_0 = 1 + (\tilde{k} + k_0)(N - (p + q - 1)) \tag{1}$$

with $\tilde{k} = \left\lfloor \frac{e\tilde{d}-1}{N+1} \right\rfloor$ and $\tilde{k} = k^* - k_0$. Moreover, since p and q are assumed to be balanced, we have $p + q \leqslant 3\sqrt{N}$. Hence, as shown in [2], it follows that

$$|k_0| = |k^* - \tilde{k}| \leqslant \left| \frac{ed^* - 1}{\phi(N)} - \frac{e\tilde{d} - 1}{N + 1} \right| \leqslant \frac{e}{\phi(N)} \left(|d_0| + 3N^{-\frac{1}{2}} \tilde{d} \right) .$$



(a) $e = 2^{16} + 1 \ \sim N^0$ (b) $e \sim N^{1/4}$ (c) $e \sim N^{1/2}$

Fig. 1. Graphical representations of the results of Corollary 1

We define δ such that $|d_0| \leqslant N^\delta$. Assuming that $\delta \leqslant \beta - \frac{1}{2}$, the conditions $e \sim N^\alpha$, $\phi(N) \sim N$, $|d_0| \leqslant N^\delta$, $\tilde{d} \sim N^\beta$ immediately yield, up to some small error factor ϵ, the bound

$$|k_0| \leqslant N^{\alpha-1+\max(\delta,-\frac{1}{2}+\beta)} = N^{\alpha+\beta-\frac{3}{2}} \ . \tag{2}$$

3.2 Trivariate approach: $\boldsymbol{\beta \geqslant 3/2 - \alpha}$

Equation (1) yields the trivariate polynomial

$$f(x, y, z) = (e\tilde{d} - 1 - \tilde{k}N) + ex - yN + \tilde{k}z + yz \,,$$

of which $(x_0, y_0, z_0) = (d_0, k_0, p + q - 1)$ is a root. Furthermore, up to a small error factor ϵ, we have the upper bounds

$$|d_0| \leqslant X = N^\delta \,, \quad |k_0| \leqslant Y = N^{\alpha+\beta-\frac{3}{2}} \,, \quad \text{and} \quad |p + q - 1| \leqslant Z = N^{\frac{1}{2}} \ .$$

We now apply the strategy of Jochemsz and May [16]. The goal is to maximize δ with respect to β, when α is a fixed value. Define two integers m and t. As sketched in § 2.2, the set S describing the monomials used for the shift polynomials contains the monomials of f^{m-1}, and the set M is defined as the set of monomials of $x^i y^j z^k \, f(x, y, z)$ with $x^i y^j z^k \in S$. Since Y is much smaller than X and Z for $\alpha + \beta < 3/2 + \delta$ (a *posteriori* we shall see that it was the good choice to make), we use extra-shifts on y. Therefore, we get

$$x^i y^j z^k \in S \iff \begin{cases} i = 0, \ldots, m-1 \\ j = 0, \ldots, m-1-i+t \\ k = 0, \ldots, m-1-i \end{cases}$$

and

$$x^i y^j z^k \in M \iff \begin{cases} i = 0, \ldots, m \\ j = 0, \ldots, m-i+t \\ k = 0, \ldots, m-i \end{cases} \ .$$

The parameter t has to be optimized with respect to m in order to maximize δ.

From the discussion in § 2.2, we know that two polynomials sharing the root (x_0, y_0, z_0) can be computed thanks to LLL algorithm as long as $X^{s_x} Y^{s_y} Z^{s_z} < W^s$ with the notation of § 2.2. First notice that, up to a small error factor ϵ, we have $W = N^{\alpha+\beta-\frac{1}{2}}$. Now defining τ such that $t = \tau m$, we obtain

$$\begin{cases} s = \left(\frac{1}{3} + \frac{1}{2}\tau\right) m^3 & + o(m^3) \\ s_x = \left(\frac{1}{3} + \frac{1}{2}\tau\right) m^3 & + o(m^3) \\ s_y = \left(\frac{1}{2} + \tau + \frac{1}{2}\tau^2\right) m^3 & + o(m^3) \\ s_z = \left(\frac{1}{2} + \frac{1}{2}\tau\right) m^3 & + o(m^3) \end{cases} .$$

In order to get the asymptotic bound, we let m grow to infinity and all the lower-order terms contribute to some error factor ϵ. The latter equation then becomes $X^{2+3\tau} Y^{3+6\tau+3\tau^2} Z^{3+3\tau} < W^{2+3\tau}$. If we substitute the values for the upper bounds, we get:

$$\delta(2+3\tau) + \left(\alpha+\beta-\tfrac{3}{2}\right)(3+6\tau+3\tau^2) + \tfrac{1}{2}(3+3\tau) < \left(\alpha+\beta-\tfrac{1}{2}\right)(2+3\tau) \ .$$

The optimal value for τ is given by $\frac{1-\frac{\alpha}{2}-\frac{\beta}{2}-\frac{\delta}{2}}{\alpha+\beta-\frac{3}{2}}$ (valid until $\alpha+\beta+\delta \leqslant 2$) and leads to

$$\delta < \frac{\alpha+\beta-\sqrt{4(\alpha+\beta)^2 - 6(\alpha+\beta)}}{3} - \epsilon \ .$$

3.3 Bivariate approach: $\beta < 3/2 - \alpha$

Analogously to what was done in the previous section, a lattice reduction leads directly to the bound $\delta < \alpha + \beta - 1 - \epsilon$.[4] However, this latter bound can simply be obtained with a straightforward argument. As $k^* \sim N^{\alpha+\beta-1}$, we can consider the bivariate polynomial

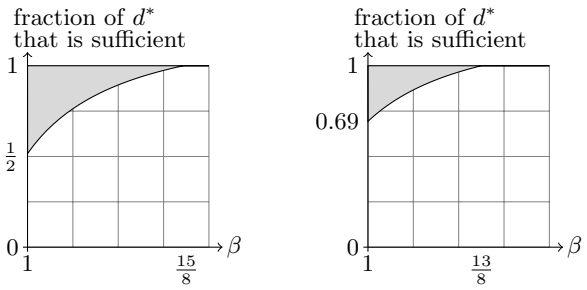
$$f(x, y) = (e\tilde{d} - 1 - k^* N) + ex + k^* y$$

modulo k^*. Indeed, when $\delta < \alpha + \beta - 1$ holds, we immediately obtain the value of d_0 over the integers as $d_0 = x_0 \bmod k^* = \frac{1-e\tilde{d}}{e} \bmod k^*$.

4 Key Recovery from Known LSBs

We now assume that the attacker succeeded to recover the least significant bits (LSBs) of $d^* = d + \ell\phi(N)$. More generally, we write $d^* = d_l + d_0 M$, where d_l is known to the attacker, together with its higher bound $M = N^\mu$, but d_0 is unknown. (In the particular case of known LSBs, M is a power of two.) We prove the following theorem:

[4] Due to the shape of the bivariate polynomial (linear in each variable), no extra-shift is necessary.



(a) $e = 2^{16} + 1 \sim N^0$ (b) $e \sim N^{1/4}$

Fig. 2. Graphical representation of the results of Corollary 2

Theorem 2. *With the previous notations, suppose that $e \sim N^\alpha$ and $d^* \sim N^\beta$. Then under Assumption 1 and up to a small error factor ϵ, there exists, for sufficiently large N, a polynomial-time algorithm that computes all of d^*, provided that*

$$\mu \geqslant \left(\frac{(6\beta - 5) + 2\sqrt{6\alpha + (6\beta - 5)}}{6} - \epsilon \right) .$$

The proof of this theorem can be seen as an application of the strategy offered in [12]. We omit it due to space limitations.

We then obtain:

Corollary 2. *Using the notation of Theorem 2, recovering a fraction of d^* larger than*

$$\frac{(6\beta - 5) + 2\sqrt{6\alpha + (6\beta - 5)}}{6\beta} .$$

is sufficient to recover the entire private exponent d^* in polynomial time.

Proof (of Corollary 2). This immediately follows from Theorem 2 by noting that the fraction of d^* corresponding to the unknown d_0 is given by $\frac{\mu}{\beta}$. □

A graphical interpretation of Corollary 2, representing the fraction of d^* required to recover the entire exponent d^* in polynomial time, is depicted in Fig. 2.

5 Practical Experiments

The attacks described in Section 3 and 4 were implemented with the SAGE computer-algebra system [29] using Shoup's NTL [26], and run on a 2.8 GHz Intel Core i5 running Mac OS X 10.7.2. In all the listed experiments, we were able to recover the factorization of N, i.e., in each case, Assumption 1 held. One can notice that our attacks are much better than predicted (when computing the theoretical value with the used values of m and t). This difference between theoretical results and practical ones is studied in [14, 25].

Table 1. Experimental results for the attack on MSBs with a 1000-bit N

(a) $e = 65537 \sim N^0$

β	δ	Γ	Parameters	LLL
1.501	0.39	0.475	$m = 4, t = 3$ dim $= 40$	42 sec
1.55	0.34	0.331	$m = 3, t = 2$ dim $= 24$	4 sec
1.60	0.25	0.267	$m = 2, t = 2$ dim $= 15$	1 sec
1.65	0.20	0.218	$m = 3, t = 3$ dim $= 28$	20 sec
1.70	0.10	0.178	$m = 6, t = 1$ dim $= 56$	27 min

(b) $e \sim N^{1/2}$

β	δ	Γ	Parameters	LLL
1.001	0.41	0.475	$m = 4, t = 3$ dim $= 40$	57 sec
1.05	0.29	0.331	$m = 4, t = 3$ dim $= 40$	1 min 45 sec
1.10	0.23	0.267	$m = 3, t = 3$ dim $= 28$	13 sec
1.15	0.20	0.218	$m = 4, t = 2$ dim $= 35$	1 min 19 sec
1.20	0.17	0.178	$m = 4, t = 2$ dim $= 35$	1 min 43 sec

5.1 Results for attack on MSBs

As an illustration, consider an RSA application making use of a 2048-bit modulus N and public exponent e. Further, in order to prevent DPA-type attacks, assume that a 128-bit random multiple ℓ of $\phi(N)$ is added to d, defining the private exponent $d^* = d + \ell\phi(N)$. Thus, $\beta = \frac{2048+128}{2048} = 1.06$.

Consider the following practical settings:

Case 1: $e = 65537$, i.e. $\alpha \sim 0$. From Corollary 1, it follows that it suffices to recover the $1/\beta = 94\%$ of d^* rather than all of it, allowing 128 bits of d^* to remain unknown. However, a practical implementation does not require lattice reduction (see § 3.3) and verifying the latter result is easy.

Case 2: $e \sim N^{1/2}$, i.e. $\alpha \sim 1/2$. Corollary 1 then tells us that it suffices to recover the 71% of d^* rather than all of it, theoretically allowing 640 bits of d^* to remain unknown.

A practical implementation with $\delta = 0.19$ (i.e. $0.19 \times 2048 = 389$ bits of d^* unknown), and parameters $m = 7, t = 1$ (dim $= 72$), allowed us to recover the 389 unknown bits of d^* in 5 hours and 48 minutes.[5]

Since the bounds of the algorithms do not depend of the length of the modulus N, all the following experiments for this attack were performed with a 1000-bit N, and three different values for the public exponent, $e = 2^{16} + 1 = 65537$ and $e \sim N^{1/2}$.

For every β value between 1 and $2-\alpha$, we looked for the bigger δ that gave us enough small vectors to recover the root (x_0, y_0, z_0). We tried for each δ different values of $m \geqslant 2$ and $t \geqslant 1$. The results are presented in Table 1. In our tests, the bound δ given in the table is reached by d_0, and the Γ-column gives the asymptotic bound which is reached when the lattice dimension goes to infinity.

[5] Notice that one can get closer to the theoretical bound by increasing the size of the lattice (at the expense of an increased running time: for $\delta = 0.18$, with parameters $m = 4$, $t = 1$, dim $= 30$, the 368 unknown bits were recovered in 2 minutes).

Table 2. Experimental results for the attack on LSBs with a 1000-bit N

(a) $e = 2^{16} + 1 \sim N^0$

β	μ	Γ	Parameters	LLL
1.01	0.58	0.535	$m = 2, t = 1$ dim $= 16$	1 sec
1.10	0.76	0.700	$m = 2, t = 1$ dim $= 16$	1 sec
1.20	0.96	0.871	$m = 2, t = 1$ dim $= 16$	1 sec
1.30	1.16	1.033	$m = 2, t = 1$ dim $= 16$	1 sec

(b) $e \sim N^{1/4}$

β	μ	Γ	Parameters	LLL
1.01	0.77	0.710	$m = 2, t = 1$ dim $= 16$	1 sec
1.10	0.92	0.854	$m = 4, t = 1$ dim $= 50$	20 sec
1.20	1.08	1.008	$m = 5, t = 1$ dim $= 77$	2 min 1 sec
1.30	1.26	1.158	$m = 4, t = 1$ dim $= 50$	22 sec

5.2 Results for attack on LSBs

All experiments for this attack were conducted with a 1000-bit N, and two different values for the public exponent, $e = 2^{16} + 1 = 65537$ and $e \sim N^{1/4}$.

For every β value between 1 and $15/8 - \alpha$, we looked for the smaller μ that gave us enough small vectors to recover the root (x_0, y_0, z_0). We tried for each μ different values of $m \geqslant 2$ and $t \geqslant 1$. The results are presented in Table 2.

In our tests, the bound μ given in the table is reached by d_l and the Γ-column gives the asymptotic bound which is reached when the lattice dimension goes to infinity.

6 Conclusion

In this paper we established sufficient conditions to successfully mount partial key exposure attacks on RSA. Unlike previous works, we focused on the practical setting of a private exponent d larger than the modulus N. We derived theoretical bounds that were validated through numerical experiments for various parameter sets. Our results illustrated once more the importance of careful implementation.

Our work raises several open issues. For the interested reader, here some possible venues for further research. Is it possible to derive bounds when $\beta \geqslant 2$? Is it possible to mount a key recovery attack when random key bits of d^* are exposed? This problem naturally finds applications in so-called cold-boot attacks; see [13, 24]. Is it possible to extend the results to CRT implementations for arbitrary values for ℓ and/or e?[6] How does the use of unbalanced primes modify the attack [1]?

References

1. Bleichenbacher, D., May, A.: New attacks on RSA with small secret CRT-exponents. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) Public Key

[6] One can apply the same strategy as above to mount a partial key recovery attack on LSBs for a low public-exponent e.

Cryptography − PKC 2006. Lecture Notes in Computer Science, vol. 3958, pp. 1–13. Springer-Verlag (2006)
2. Blömer, J., May, A.: New partial key exposure attacks on RSA. In: Boneh, D. (ed.) Advances in Cryptology − CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 27–43. Springer-Verlag (2003)
3. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Transactions on Information Theory 46(4), 1339–1349 (2000), extended abstract in Proc. of EUROCRYPT '98
4. Boneh, D., Durfee, G., Frankel, Y.: An attack on RSA given a small fraction of the private key bits. In: Ohta, K., Pei, D. (eds.) Advances in Cryptology − ASIACRYPT '98. Lecture Notes in Computer Science, vol. 1514, pp. 25–34. Springer-Verlag (1998)
5. Cohen, G.D., Lobstein, A., Naccache, D., Zémor, G.: How to improve an exponentiation black-box. In: Nyberg, K. (ed.) Advances in Cryptology − EUROCRYPT '98. Lecture Notes in Computer Science, vol. 1403, pp. 211–220. Springer-Verlag (1998)
6. Coppersmith, D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Maurer, U.M. (ed.) Advances in Cryptology − EUROCRYPT '96. Lecture Notes in Computer Science, vol. 1070, pp. 178–189. Springer-Verlag (1996)
7. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) Advances in Cryptology − EUROCRYPT '96. Lecture Notes in Computer Science, vol. 1070, pp. 155–165. Springer-Verlag (1996)
8. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10(4), 233–260 (1997)
9. Coron, J.S.: Resistance against differential power analysis for elliptic curve cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems (CHES '99). Lecture Notes in Computer Science, vol. 1717, pp. 292–302. Springer-Verlag (1999)
10. Coron, J.S.: Finding small roots of bivariate integer polynomial equations revisited. In: Cachin, C., Camenisch, J. (eds.) Advances in Cryptology − EUROCRYPT 2004. Lecture Notes in Computer Science, vol. 3027, pp. 492–505. Springer-Verlag (2004)
11. Coron, J.S.: Finding small roots of bivariate integer polynomial equations: A direct approach. In: Menezes, A. (ed.) Advances in Cryptology − CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 379–394. Springer-Verlag (2007)
12. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial key exposure attacks on RSA up to full size exponents. In: Cramer, R. (ed.) Advances in Cryptology − EUROCRYPT 2005. Lecture Notes in Computer Science, vol. 3494, pp. 371–386. Springer-Verlag (2005)
13. Heninger, N., Shacham, H.: Reconstructing RSA private keys from random key bits. In: Halevi, S. (ed.) Advances in Cryptology − CRYPTO 2009. Lecture Notes in Computer Science, vol. 5677, pp. 1–17. Springer-Verlag (2009)
14. Herrmann, M., May, A.: Maximizing small root bounds by linearization and applications to small secret exponent RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) Public Key Cryptography − PKC 2010. Lecture Notes in Computer Science, vol. 6056, pp. 53–69. Springer-Verlag (2010)
15. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) Cryptography and Coding. Lecture Notes in Computer Science, vol. 1355, pp. 131–142. Springer-Verlag (1997)

16. Jochemsz, E., May, A.: A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In: Lai, X., Chen, K. (eds.) Advances in Cryptology − ASIACRYPT 2006. Lecture Notes in Computer Science, vol. 4284, pp. 267–282. Springer-Verlag (2006)
17. Jochemsz, E., May, A.: A polynomial time attack on RSA with private CRT-exponents smaller than $N^{0.073}$. In: Menezes, A. (ed.) Advances in Cryptology − CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 395–411. Springer-Verlag (2007)
18. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) Advances in Cryptology − CRYPTO '99. Lecture Notes in Computer Science, vol. 1666, pp. 388–397. Springer-Verlag (1999)
19. Kocher, P., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. Journal of Cryptographic Engineeering 1(1), 5–27 (2011)
20. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) Advances in Cryptology − CRYPTO '96. Lecture Notes in Computer Science, vol. 1109, pp. 104–113. Springer-Verlag (1996)
21. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)
22. May, A.: New RSA Vulnerabilities Using Lattice Reduction Methods. Ph.D. thesis, University of Paderborn (2003)
23. Miller, G.L.: Riemann's hypothesis and tests for primality. Journal of Computer and System Sciences 13(3), 300–317 (1976)
24. Sarkar, S.: Partial key exposure: Generalized framework to attack RSA. In: Bernstein, D.J., Chatterjee, S. (eds.) Progress in Cryptology − INDOCRYPT 2011. Lecture Notes in Computer Science, vol. 7107, pp. 76–92. Springer-Verlag (2011)
25. Sarkar, S., Sengupta, S., Maitra, S.: Partial key exposure attack on RSA - improvements for limited lattice dimensions. In: Gong, G., Gupta, K.C. (eds.) Progress in Cryptology − INDOCRYPT 2010. Lecture Notes in Computer Science, vol. 6498, pp. 2–16. Springer-Verlag (2010)
26. Shoup, V.: Number Theory Library (Version 5.5.2). A library for doing Number Theory (2011), `http://www.shoup.net/ntl`
27. Simmons, G.J.: The prisoners' problem and the subliminal channel. In: Chaum, D. (ed.) Advances in Cryptology, Proceedings of CRYPTO '83. pp. 51–67. Plenum Press (1984)
28. Simmons, G.J.: The subliminal channel and digital signature. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) Advances in Cryptology, Proceedings of EUROCRYPT 84. Lecture Notes in Computer Science, vol. 209, pp. 364–378. Springer-Verlag (1984)
29. Stein, W.A., et al.: Sage Mathematics Software (Version 4.7). The Sage Development Team (2011), `http://www.sagemath.org`
30. Wiener, M.J.: Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory 36(3), 553–558 (1990)
31. Young, A., Yung, M.: Malicious Cryptography: Exposing Cryptovirology. John Wiley & Sons (2004)