A Polynomial Time Attack on RSA with Private CRT-Exponents Smaller Than $N^{0.073}$

Ellen Jochemsz[1]** and Alexander May[2]

[1] Department of Mathematics and Computer Science,
TU Eindhoven, 5600 MB Eindhoven, the Netherlands
e.jochemsz@tue.nl
[2] Faculty of Computer Science
TU Darmstadt, 64289 Darmstadt, Germany
may@informatik.tu-darmstadt.de

Abstract. Wiener's famous attack on RSA with $d < N^{0.25}$ shows that using a small d for an efficient decryption process makes RSA completely insecure. As an alternative, Wiener proposed to use the Chinese Remainder Theorem in the decryption phase, where $d_p = d \mod (p-1)$ and $d_q = d \mod (q-1)$ are chosen significantly smaller than p and q. The parameters d_p, d_q are called private CRT-exponents. Since Wiener's proposal in 1990, it has been a challenging open question whether there exists a polynomial time attack on small private CRT-exponents. In this paper, we give an affirmative answer to this question, and show that a polynomial time attack exists if d_p and d_q are smaller than $N^{0.073}$.

Keywords: RSA, CRT, cryptanalysis, small exponents, Coppersmith's method.

1 Introduction

In the RSA cryptosystem, the public modulus $N = pq$ is a product of two primes of the same bitsize. The public and private exponent e and d satisfy

$$ed = 1 \mod (p-1)(q-1).$$

In many applications of RSA, either e or d is chosen to be small, for efficient modular exponentiation in the encryption/verifying or in the decryption/signing phase. It is well-known that it is dangerous to choose a small private exponent, since Wiener [22] showed that the RSA scheme is insecure if $d < N^{0.25}$, which was extended to $d < N^{0.292}$ by Boneh and Durfee [4].

** The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

As an alternative approach, Wiener proposed to use the Chinese Remainder Theorem (CRT) for decryption/signing as described by Quisquater and Couvreur in [18], and to use small private CRT-exponents instead of a small private exponent. In that case, the public exponent e and private CRT-exponents d_p and d_q satisfy $ed_p = 1 \bmod (p-1)$ and $ed_q = 1 \bmod (q-1)$. To obtain a fast decryption/signing phase, d_p and d_q are chosen significantly smaller than p and q. In time-critical applications, for instance for signing procedures on smart-cards, this technique is especially useful. Whether there exists a polynomial time attack on this RSA-CRT system with small d_p and d_q has been a challenging open question since Wiener's work (see also the comments in Boneh-Durfee [4], the STORK roadmap [19], and the ECRYPT document on the hardness of the main computational problems in cryptography [9]).

So far, the best attack on this variant is a square-root attack [3] that enables an adversary to factor N in time and space $\tilde{O}(\min\{\sqrt{d_p}, \sqrt{d_q}\})$, which is exponential in the bitsize of d_p and d_q. All other attacks on RSA with small private CRT-exponents can be divided in two categories.

First, there are attacks on the special case where p and q are 'unbalanced' (not of the same bitsize). May [16] described two attacks that work up to a smallest prime factor of $N^{0.382}$. Recently, Bleichenbacher and May [2] improved this to $N^{0.468}$.

Secondly, there are attacks on a special case where not only d_p and d_q, but also e is chosen to be small. Galbraith, Heneghan and McKee [10] and Sun and Wu [20] have made proposals to use RSA-CRT in a way that 'balances' the cost of encryption and decryption by forcing both e and d_p, d_q to be small. In these articles, several attacks are described, after which the authors propose parameters that are not affected by these attacks. Bleichenbacher and May [2] in turn described a new attack on RSA-CRT with balanced exponents, forcing Galbraith, Heneghan, McKee and Sun, Wu to revise their parameter suggestions in [11] and [21], respectively.

However, the attacks in both categories are not applicable in the standard RSA case with small CRT-exponents d_p and d_q, that is, when p and q are balanced and e is full size. In this paper, we describe a way to extend one of the attacks of Bleichenbacher, May [2] such that it also works in the standard RSA-CRT case. This leads to the first polynomial time attack on standard RSA with small private CRT-exponents. More precisely, we present the following result.

Theorem 1 (RSA-CRT with Small d_p, d_q). *Under a well-known heuristic assumption (as described in Section 6), for every $\epsilon > 0$ and sufficiently large n, the following holds:*
Let $N = pq$ be an n-bit RSA modulus, and p, q primes of bitsize $\frac{n}{2}$. Let $e < \phi(N)$, $d_p < p-1$, and $d_q < q-1$ be the public exponent and private CRT-exponents, satisfying $ed_p \equiv 1 \bmod (p-1)$ and $ed_q \equiv 1 \bmod (q-1)$. Let bitsize$(d_p) \leq \delta n$ and bitsize$(d_q) \leq \delta n$. Then N can be factored in time polynomial in $\log(N)$ provided that

$$\delta < 0.0734 - \epsilon.$$

The rest of the paper is organized as follows. In Section 2, we give a brief introduction to Coppersmith's lattice-based method for finding small roots of polynomials [5]. In Section 3, we recall the Bleichenbacher-May attack [2]. In Section 4, we show how an extension of the attack leads to our new attack on standard RSA-CRT with $\delta < 0.0734 - \epsilon$. Furthermore, we generalize our bound to public exponents e of arbitrary size, and show that this leads to a polynomial time attack on one of the revised parameter choices in [21]. In Section 5, we explain in detail how we use Coppersmith's original method for the implementation of the attack. In Section 6, we discuss the only heuristic part of the attack, namely how to retrieve a common root from a number of polynomials. We conclude in Section 7 by giving experimental data for our attack.

2 Finding Small Roots of Polynomials

Many attacks in RSA cryptanalysis use a similar technique, which originated from Coppersmith's work on finding small roots of polynomials [5]. In essence, the attack starts with a polynomial equation in some of the unknowns of the RSA variant, such as p, q, d, or d_p and d_q in the case of RSA-CRT. An example is the usual RSA equation

$$ed = 1 + k(N + 1 - (p + q)),$$

with the unknowns d, k, p, and q.

Such an equation yields a polynomial f which has a certain root that an attacker wishes to find. In the example, the polynomial

$$f(x_1, x_2, x_3) = ex_1 - 1 - x_2(N + 1 - x_3)$$

has the root $(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}) = (d, k, p + q)$. Finding the root is equivalent to factoring N, since p, q can be computed from $p + q$ using $N = pq$. The goal is to derive a polynomial time attack provided that the size of the root is below a certain bound.

In our new attack on standard RSA-CRT (Section 4), our goal is to find a root of a four-variate polynomial $f(x_1, x_2, x_3, x_4)$. We follow the strategy of Jochemsz and May [13], that we will sketch here.

Let $(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)})$ be a root of the polynomial $f(x_1, x_2, x_3, x_4)$ that is small in the sense that $|x_1^{(0)}| < X_1,\ |x_2^{(0)}| < X_2,\ |x_3^{(0)}| < X_3,\ |x_4^{(0)}| < X_4$, for some known upper bounds X_j, for $j = 1, \ldots, 4$. Moreover, we define W as the maximal absolute coefficient of $f(x_1X_1, x_2X_2, x_3X_3, x_4X_4)$. That is, $W := \|f(x_1X_1, x_2X_2, x_3X_3, x_4X_4)\|_\infty$, where $\|f(x_1, x_2, x_3, x_4)\|_\infty = \max |a_{i_1 i_2 i_3 i_4}|$ for a polynomial $f(x_1, x_2, x_3, x_4) = \sum a_{i_1 i_2 i_3 i_4} x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4}$.

A basis B of a lattice L is defined via so-called shift polynomials of the form $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} f(x_1, x_2, x_3, x_4)$. The choice of the combinations $\{i_1, i_2, i_3, i_4\}$ that are used is described by a set S. The set M then consists of all monomials that appear in the shift polynomials. The choice of S is crucial and depends on the monomials that appear in f. We will give the precise definition of S in Section 4 for our specific polynomial.

Then, LLL-reduction [15] is performed on B to find small vectors in the lattice L. From a result of [13], we know that under the condition

$$X_1^{s_1} X_2^{s_2} X_3^{s_3} X_4^{s_4} < W^s, \text{ for } s_j = \sum_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S} i_j \quad \text{and } s = |S|, \tag{1}$$

the first vectors in the reduced basis are small enough to ensure that we find a list $f_0, \ldots, f_\ell$ of at least three polynomials that all have the root $(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)})$ over the integers. The polynomials $\{f, f_0, \ldots, f_\ell\}$ will reveal their common root $(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)})$ under the assumption that three variables can be eliminated from the polynomial system of equations $\{f = 0, f_0 = 0, \ldots, f_\ell = 0\}$. Resultant computations are often used for this elimination process, but we choose to use Gröbner Bases, as we will explain in Section 6. Experiments must be done to verify that the elimination assumption holds in practice.

3 The Bleichenbacher-May Attack

In [2], Bleichenbacher and May describe two new attacks on RSA-CRT. One of them is meant for the case that both e and d_p and d_q are chosen to be smaller than in standard RSA-CRT. For notation, we use $e = N^\alpha$, $d_p < N^\delta$, and $d_q < N^\delta$ for some $\alpha \in [0, 1]$ and $\delta \in [0, \frac{1}{2}]$. Clearly, if an attack on this so called 'balanced' RSA works in the case $\alpha = 1$, then it threatens the security of standard RSA with small private CRT-exponents.

The attack of Bleichenbacher and May uses a lattice of dimension 3. The attack works whenever $\delta < \min\{\frac{1}{4}\ ,\ \frac{2}{5} - \frac{2}{5}\ \alpha\}$, and therefore gives no result in the case $\alpha = 1$. However, we present a generalization of the attack for higher dimensional lattices that is applicable also for $\alpha = 1$. To explain our new attack, we first describe the basics of the BM-attack [2].

Bleichenbacher and May start with the two RSA-CRT equations $ed_p = 1 + k(p-1)$ and $ed_q = 1 + l(q-1)$, and rewrite these as

$$ed_p + k - 1 = kp \quad \text{and} \quad ed_q + l - 1 = lq.$$

Multiplying the two equations yields

$$e^2 d_p d_q + ed_p(l-1) + ed_q(k-1) - (N-1)kl - (k+l-1) = 0.$$

This can be transformed into the linear equation $e^2 x_1 + e x_2 - (N-1)x_3 - x_4 = 0$, if we substitute $x_1 = d_p d_q$, $x_2 = d_p(l-1) + d_q(k-1)$, $x_3 = kl$, $x_4 = k + l - 1$.

The given linear equation leads directly to a lattice attack with a lattice of dimension 3. This attack works provided that $\delta < \min\{\frac{1}{4}\ ,\ \frac{2}{5} - \frac{2}{5}\ \alpha\}$.

Although linearization of an equation makes the analysis easier and keeps the lattice dimension small, better results can sometimes be obtained by using a non-linear polynomial equation directly. In the next section, we will pursue this approach and use a polynomial with the variables $x_1, \ldots, x_4$ corresponding to d_p, d_q, k, and l, respectively.

4 The New Attack on RSA-CRT

The equation we introduced in the previous section

$$e^2 d_p d_q + e d_p(l-1) + e d_q(k-1) - (N-1)kl - (k+l-1) = 0$$

yields a polynomial $f(x_1, x_2, x_3, x_4) = e^2 x_1 x_2 + e x_1 x_4 - e x_1 + e x_2 x_3 - e x_2 - (N-1) x_3 x_4 - x_3 - x_4 + 1$ with monomials $1, x_1, x_2, x_3, x_4, x_1x_2, x_1x_4, x_2x_3, x_3x_4$ and a small root

$$(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)}) = (d_p, d_q, k, l), \text{ with } \begin{cases} |x_1^{(0)}| < X_1 = N^{\delta}, \\ |x_2^{(0)}| < X_2 = N^{\delta}, \\ |x_3^{(0)}| < X_3 = N^{\alpha+\delta-\frac{1}{2}}, \\ |x_4^{(0)}| < X_4 = N^{\alpha+\delta-\frac{1}{2}}. \end{cases}$$

We will follow the strategy for finding small integer roots of Jochemsz and May [13] as sketched in Section 2, to analyze which attack bound corresponds to this polynomial f.

In the basic strategy of [13], the set S that describes which monomials $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4}$ are used for the shift polynomials, is simply the set that contains all monomials of f^{m-1} for a given integer m. The set M is defined as the set of all monomials that appear in $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} f(x_1, x_2, x_3, x_4)$, with $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S$. Since f has a non-zero constant coefficient, all monomials of S are included in M. More precisely, S and M can be described as

$$x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S \Leftrightarrow \begin{cases} i_1 = 0, \ldots, m-1-i_3, \\ i_2 = 0, \ldots, m-1-i_4, \\ i_3 = 0, \ldots, m-1, \\ i_4 = 0, \ldots, m-1, \end{cases}$$

$$x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \Leftrightarrow \begin{cases} i_1 = 0, \ldots, m-i_3, \\ i_2 = 0, \ldots, m-i_4, \\ i_3 = 0, \ldots, m, \\ i_4 = 0, \ldots, m. \end{cases}$$

However, in [13] it is also advised to explore the possibility of extra shifts of one or more variables. Since X_1 and X_2 are significantly smaller than X_3 and X_4 for $\alpha > \frac{1}{2}$, we find that the attack bound is superior for $\alpha = 1$ if we use extra shifts of x_1 and x_2. Therefore, we take

$$x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S \Leftrightarrow \begin{cases} i_1 = 0, \ldots, m-1-i_3+t, \\ i_2 = 0, \ldots, m-1-i_4+t, \\ i_3 = 0, \ldots, m-1, \\ i_4 = 0, \ldots, m-1, \end{cases}$$

$$x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \Leftrightarrow \begin{cases} i_1 = 0, \ldots, m-i_3+t, \\ i_2 = 0, \ldots, m-i_4+t, \\ i_3 = 0, \ldots, m, \\ i_4 = 0, \ldots, m, \end{cases}$$

for some t that has to be optimized as a function of m and α.

Our goal is to find at least three polynomials f_0, f_1, f_2 that share the root $(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)})$ over the integers. From Section 2 we know that these polynomials can be computed by lattice reduction techniques as long as

$$X_1^{s_1} X_2^{s_2} X_3^{s_3} X_4^{s_4} < W^s, \text{ for } s_j = \sum_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S} i_j \quad \text{and } s = |S|.$$

For a given integer m and $t = \tau m$, our last definition of S and M yields the bound

$$(X_1 X_2)^{(\frac{5}{12} + \frac{5}{3}\tau + \frac{9}{4}\tau^2 + \tau^3) m^4 + o(m^4)} (X_3 X_4)^{(\frac{5}{12} + \frac{5}{3}\tau + \frac{3}{2}\tau^2) m^4 + o(m^4)} < W^{(\frac{1}{4} + \tau + \tau^2) m^4 + o(m^4)}.$$

To obtain the asymptotic bound, we let m grow to infinity and let all terms of order $o(m^4)$ contribute to some error term ϵ. If we substitute the values for X_1, X_2, X_3, X_4, W, we obtain

$$\left(\tfrac{5}{12} + \tfrac{5}{3}\tau + \tfrac{9}{4}\tau^2 + \tau^3\right) \cdot 2\delta + \left(\tfrac{5}{12} + \tfrac{5}{3}\tau + \tfrac{3}{2}\tau^2\right) \cdot (2\alpha + 2\delta - 1) < \left(\tfrac{1}{4} + \tau + \tau^2\right) \cdot (2\alpha + 2\delta),$$

which leads to

$$\delta < \frac{5 - 4\alpha + 20\tau - 16\alpha\tau + 18\tau^2 - 12\alpha\tau^2}{14 + 56\tau + 66\tau^2 + 24\tau^3} - \epsilon.$$

For $\alpha = 1$, we find an optimal value of $\tau \approx 0.381788$, and we get

$$\delta < 0.0734 - \epsilon.$$

Hence, for a 1024-bit modulus, d_p and d_q are in the attack space if they are less then 75 bits. Analogously, for a 2048-bit modulus, d_p and d_q are in the attack space if they are at most 150 bits.

4.1 Extending the Attack to Other Values of α

In Section 4, we assumed that $x_1^{(0)}, x_2^{(0)}$ are smaller than $x_3^{(0)}, x_4^{(0)}$, t.i. $\alpha \geq \frac{1}{2}$. For $\alpha < \frac{1}{2}$, symmetrically one uses extra x_3 and x_4-shifts instead of extra x_1 and x_2-shifts. Because of the symmetry, one can immediately see that the attack bound is

$$(X_1 X_2)^{(\frac{5}{12} + \frac{5}{3}\tau + \frac{3}{2}\tau^2) m^4 + o(m^4)} (X_3 X_4)^{(\frac{5}{12} + \frac{5}{3}\tau + \frac{9}{4}\tau^2 + \tau^3) m^4 + o(m^4)} < W^{(\frac{1}{4} + \tau + \tau^2) m^4 + o(m^4)}.$$

The above bound leads to

$$\delta < \frac{5 - 4\alpha + 20\tau - 16\alpha\tau + 27\tau^2 - 30\alpha\tau^2 + 12\tau^3 - 24\alpha\tau^3}{14 + 56\tau + 66\tau^2 + 24\tau^3} - \epsilon.$$

Note that this bound only holds for $\alpha + \delta > \frac{1}{2}$, since we assume that the values of k and l are unknown to the attacker. Both conditions are only met if $\alpha \geq \frac{1}{6}$.

However, in Section 7.1 we provide experimental evidence that our heuristic attack is successful only when $\alpha \geq \frac{1}{4}$.

In the revised paper by Sun, Hinek, Wu [21], the authors propose as new parameters $\{\alpha = 0.577, \delta = 0.186\}$. For this choice, we find the bound $\delta < 0.192$, which breaks the new proposal in polynomial time.

5 Implementation Using Coppersmith's Original Method

Although we have derived our attack bound from the strategy of Jochemsz, May [13], we deviate from their strategy for the implementation of the attack. Basically, we make use of Coppersmith's original technique [5] instead of Coron's reformulation [6]. This does not change the asymptotic bound of the attack, but it has a major practical advantage. Namely, the lattices used in the attacks are high-dimensional, and Coppersmith's original method requires only the reduction of a lower-dimensional sublattice[1]. Since the LLL-process is the most costly factor in our attack, this leads to a significant improvement in practice. Furthermore, we slightly adapt Coppersmith's original method such that we directly obtain triangular lattice bases, which simplifies the determinant calculations.

So let us first explain how to apply Coppersmith's technique for our attack. We introduce the shift polynomials

$$g_{i_1 i_2 i_3 i_4}(x_1, x_2, x_3, x_4) = x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} f(x_1, x_2, x_3, x_4),$$

for $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S$ for a set of monomials S, as specified in Section 4. As before, we define the set M as the set of all monomials that appear in the shift polynomials. We use the notation $s = |S|$ for the total number of shifts and $d = |M| - |S|$ for the difference of the number of monomials and the number of shifts. Notice that the maximal coefficient of $f(x_1X_1, x_2X_2, x_3X_3, x_4X_4)$ is $e^2 X_1 X_2$, and the monomial corresponding to it is $x_1 x_2$. We define S' as the set of monomials $x_1^{i_1+1} x_2^{i_2+1} x_3^{i_3} x_4^{i_4}$, for $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S$. Naturally, $|S'| = |S| = s$. We now build a $(d+s) \times (d+s)$ matrix B_1 as follows.

The upper left $d \times d$ block is diagonal, where the rows represent the monomials $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \backslash S'$. The diagonal entry of the row corresponding to $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4}$ is $(X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4})^{-1}$. The lower left $s \times d$ block contains only zeros.

The last s columns of the matrix B_1 represent the shift polynomials $g_{i_1 i_2 i_3 i_4} = x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} f$, for $x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S$. The first d rows correspond to the monomials in $M \backslash S'$, and the last s rows to the monomials of S'. The entry in the column corresponding to $g_{i_1 i_2 i_3 i_4}$ is the coefficient of the monomial in $g_{i_1 i_2 i_3 i_4}$.

This description asks for a simple example. Let us use the set S as described in Section 4 with $m = 1$ and $t = 0$, which results in the lattice basis B_1 given in

[1] In these CRYPTO'07 proceedings, a new article by Coron [7] shows how to adapt his method such that it also requires only the reduction of a sublattice instead of the reduction of the full lattice, and hence his new technique could be applied here, too.

Figure 1. We only use the polynomial $f(x_1, x_2, x_3, x_4)$ itself as a shift polynomial. Therefore, $s = 1$ and we have $d+s = 9$ monomials. The rows represent the monomials $1, x_1, x_2, x_3, x_4, x_3x_4, x_2x_3, x_1x_4, x_1x_2$ and the last column corresponds to the coefficients of these monomials in f.

$$\left(\begin{array}{cccccccc|c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & \frac{1}{X_1} & 0 & 0 & 0 & 0 & 0 & 0 & -e \\
0 & 0 & \frac{1}{X_2} & 0 & 0 & 0 & 0 & 0 & -e \\
0 & 0 & 0 & \frac{1}{X_3} & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & \frac{1}{X_4} & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & \frac{1}{X_3X_4} & 0 & 0 & 1-N \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{X_2X_3} & 0 & e \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{X_1X_4} & e \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^2
\end{array}\right)$$

Fig. 1. Matrix B_1 for the case $m = 1$, $t = 0$

In general, the determinant of the matrix B_1 is

$$\det(B_1) = \left(\prod_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S'} (X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4})^{-1} \right) \cdot (e^2)^s.$$

Let

$$\mathbf{v}(x_1, x_2, x_3, x_4) = (1, x_1, x_2, x_3, x_4, x_3x_4, x_2x_3, x_1x_4, x_1x_2).$$

Note that in our example,

$$\mathbf{v}(x_1, x_2, x_3, x_4) \cdot B_1 := (1, \tfrac{x_1}{X_1}, \tfrac{x_2}{X_2}, \tfrac{x_3}{X_3}, \tfrac{x_4}{X_4}, \tfrac{x_3x_4}{X_3X_4}, \tfrac{x_2x_3}{X_2X_3}, \tfrac{x_1x_4}{X_1X_4}, f(x_1, x_2, x_3, x_4)).$$

So,

$$\|\mathbf{v}(d_p, d_q, k, l) \cdot B_1\| = \|(1, \tfrac{d_p}{X_1}, \tfrac{d_q}{X_2}, \tfrac{k}{X_3}, \tfrac{l}{X_4}, \tfrac{kl}{X_3X_4}, \tfrac{d_qk}{X_2X_3}, \tfrac{d_pl}{X_1X_4}, 0)\| \le \sqrt{d}.$$

Since the X_j upper bound the root, there is always such a vector $\mathbf{v}$ which, if one substitutes the unknowns $\{d_p, d_q, k, l\}$ for the variables $\{x_1, x_2, x_3, x_4\}$, becomes a vector with Euclidean norm smaller than $\sqrt{d}$ after multiplication with the matrix B_1.

Let us perform a unimodular transformation U_1 on B_1 to create a matrix B_2 such that

$$B_2 = U_1 \cdot B_1 = \left(\begin{array}{c|c} A_{d\times d} & 0_{d\times s} \\ \hline A'_{s\times d} & I_{s\times s} \end{array}\right).$$

Now if the rows of B_1 form a basis of a lattice L, then the rows of B_2 form a basis of the same lattice. Moreover, the rows of

$$B_3 = \left(A_{d\times d} \mid 0_{d\times s} \right)$$

are a basis of the sublattice L_0 of L which has zeros in the last s entries. Notice that $\det(L_0) = \det(L)$. Clearly, $\mathbf{v}(d_p, d_q, k, l) \cdot B_1$ is in the lattice L_0 spanned by the rows of B_3.
Since

$$\mathbf{v}(d_p, d_q, k, l) \cdot B_1 = \mathbf{v}(d_p, d_q, k, l) U_1^{-1} B_2,$$

this means that the last s entries of $\mathbf{v}(d_p, d_q, k, l)U_1^{-1}$ must be zero. We use the notation $\lfloor \mathbf{v} \rfloor_{\text{sh}}$ for the vector $\mathbf{v}$ with its length 'shortened' to its first d entries. Then,

$$\lfloor \mathbf{v}(d_p, d_q, k, l) \cdot B_1 \rfloor_{\text{sh}} = \lfloor \mathbf{v}(d_p, d_q, k, l) U_1^{-1} B_2 \rfloor_{\text{sh}} = \lfloor \mathbf{v}(d_p, d_q, k, l) U_1^{-1} \rfloor_{\text{sh}} A.$$

Next, we reduce A using lattice basis reduction to a basis $B = U_2 A$. It follows that

$$\lfloor \mathbf{v}(d_p, d_q, k, l) \cdot B_1 \rfloor_{\text{sh}} = \lfloor \mathbf{v}(d_p, d_q, k, l) U_1^{-1} \rfloor_{\text{sh}} U_2^{-1} B.$$

We use the notation $\mathbf{v}'(d_p, d_q, k, l)$ for the vector $\lfloor \mathbf{v}(d_p, d_q, k, l)U_1^{-1} \rfloor_{\text{sh}} U_2^{-1}$, and B^* (with row vectors $\mathbf{b}_i^*$) for the basis after applying Gram-Schmidt orthogonalization to B. Now we can make three observations. Firstly, the vector $\mathbf{v}'$ is integral. This is because both matrices U_1 and U_2 have integer entries. Secondly, $\|\mathbf{v}(d_p, d_q, k, l) \cdot B_1\| < \sqrt{d}$. Thirdly, it is known [15] that the Gram-Schmidt orthogonalization of the LLL-reduced basis satisfies

$$\|\mathbf{b}_d^*\| \geq 2^{\frac{-(d-1)}{4}} \det(L)^{\frac{1}{d}}.$$

So, if we combine these three facts, we obtain that

$$\sqrt{d} \geq \|\mathbf{v}(d_p, d_q, k, l) \cdot B_1\| = \| \lfloor \mathbf{v}(d_p, d_q, k, l) \cdot B_1 \rfloor_{\text{sh}} \| = \| \mathbf{v}'(d_p, d_q, k, l) B \|$$

$$\geq | \mathbf{v}'(d_p, d_q, k, l)_d | \cdot \|\mathbf{b}_d^*\| \geq | \mathbf{v}'(d_p, d_q, k, l)_d | \cdot 2^{\frac{-(d-1)}{4}} \det(L)^{\frac{1}{d}}.$$

Since the terms $2^{\frac{-(d-1)}{4}}$ and $\sqrt{d}$ do not depend on N, we let them contribute to an error term ϵ. Thus, whenever

$$\det(L)^{\frac{1}{d}} > 1,$$

we must have $| \mathbf{v}'(d_p, d_q, k, l)_d | = 0$.

Hence, the polynomial $f_0(x_1, x_2, x_3, x_4)$ corresponding to the coefficient vector $\mathbf{v}'(x_1, x_2, x_3, x_4)_d$ contains the root (d_p, d_q, k, l) over the integers.

In Appendix A, we show that the bound $\det(L)^{\frac{1}{d}} > 1$ is equivalent to the bound (1) that was given in Section 2. Moreover, we use a result from Jutla [14] to show that the vectors $\mathbf{v}'(x_1, x_2, x_3, x_4)_{d-\ell}$, $\ell \geq 2$, yield a list of at least three polynomials $f_0, \ldots, f_\ell$ having the same root (d_p, d_q, k, l). In the next section, we show how to retrieve this root from the polynomials $f,\ f_0, \ldots, f_\ell$.

The running time of our algorithm is dominated by the time to LLL-reduce the lattice basis A. Taking the algorithm of Nguyen, Stehlé [17] this can be achieved in $\mathcal{O}(d^5(d + \log A_m) \log A_m)$, where $\log A_m$ is the maximal bitsize of an entry in A. Our lattice dimension d depends on ϵ^{-1} only, whereas the bitsize of the entries is bounded by a polynomial in $\log N$. Therefore, the construction of $f_0, \ldots, f_\ell$ can be done in time polynomial in $\log N$.

Moreover, $f_0, \ldots, f_\ell$ have a fixed degree that only depends on ϵ^{-1} and coefficients with bitsize polynomial in $\log N$. This will be important for the analysis in the following section.

6 Extracting the Common Root

Assume that we want to retrieve a common root from four polynomials f, f_0, f_1, f_2. Usually, one uses resultants to eliminate variables one by one until one obtains a univariate polynomial $w_0(x_1)$ that has $x_1^{(0)}$ as a root:

$$
\begin{array}{lll}
r_0(x_1,x_2,x_3) = \mathrm{Res}_{x_4}(f, f_0) & & \\
 & s_0(x_1,x_2) = \mathrm{Res}_{x_3}(r_0, r_1) & \\
r_1(x_1,x_2,x_3) = \mathrm{Res}_{x_4}(f, f_1) & & w_0(x_1) = \mathrm{Res}_{x_2}(r_3, r_4) \\
 & s_1(x_1,x_2) = \mathrm{Res}_{x_3}(r_1, r_2) & \\
r_2(x_1,x_2,x_3) = \mathrm{Res}_{x_4}(f, f_2) & &
\end{array}
$$

However, this method only works if the polynomials are algebraically independent. One cannot easily use more than three candidates f_j, besides repeating the scheme for different combinations. Moreover, the last resultant computation can take a significant amount of time and memory, since the degrees of the resultant polynomials grow fast. We use Gröbner Bases instead of resultant methods to extract the root. For a detailed introduction to Gröbner Bases, we refer to [8].

Suppose we have a set of polynomials $\{f, f_0, \ldots, f_\ell\}$ that have the small root $(x_1^{(0)}, \ldots, x_n^{(0)})$ in common. Then a Gröbner Basis $G := \{g_1, \ldots, g_t\}$ is a set of polynomials that preserves the set of common roots of $\{f, f_0, \ldots, f_\ell\}$. In other words, the variety of the ideal I generated by $\{g_1, \ldots, g_t\}$ is the same as the variety of the ideal generated by $\{f, f_0, \ldots, f_l\}$. The advantage of having a Gröbner Basis is that the g_i can be computed with respect to some ordering that eliminates the variables. Having such an elimination ordering, it is easy to extract the desired root.

In our experiments in Section 7 we usually found much more polynomials $f_0, \ldots, f_\ell$ than the required amount of $\ell = 2$. Therefore, we have two advantages of Gröbner Bases in comparison with resultants. First, in contrast to resultants the computation time of a Gröbner Basis usually benefits from more overdefined systems which lowers the time for extracting the root. Second, we do not have to search over all subsets of three polynomials until we find an algebraically independent one. Instead, we simply put all the polynomials in our Gröbner Basis computation. The elimination of variables can only fail if the variety $\mathbf{V}(I)$ defined by the ideal I which is generated by $\{f, f_0, \ldots, f_\ell\}$ is not zero-dimensional. Therefore, we make the following heuristic assumption for our attack.

Assumption 1: *The variety* $\boldsymbol{V}(I)$ *of the ideal* I *generated by the polynomials in the construction of Section 5 is zero-dimensional.*

Under Assumption 1, the secret root (d_p, d_q, k, l) can be derived in polynomial time, since we run a Gröbner Basis computation on polynomials of a fixed degree.

Recently, Bauer and Joux [1] made some important progress considering the heuristic involved in Coppersmith methods. Their result, for roots of trivariate polynomials, can in theory be extended to more variables. In this way, one could investigate if Assumption 1 can be replaced by a weaker assumption. In this paper, we made no efforts in this direction. Instead we verified the validity of Assumption 1 by experiments.

7 Experiments

In order to test the attack described in this paper for varying bitsizes of e and d_p, d_q we designed a key generation process similar to the one proposed by Galbraith, Heneghan, and McKee [10].

INPUT: Bitsizes n of N, αn of e, δn of d_p, d_q

(1) Choose d_p, d_q of bitsize δn.
(2) Choose k, l of bitsize $(\alpha + \delta - \frac{1}{2})n$ such that $\gcd(d_p, k) = \gcd(d_q, l) = \gcd(k, l) = 1$.
(3) Compute e using Chinese Remaindering such that

$$\left| \begin{array}{l} e = d_p^{-1} \mod k \\ e = d_q^{-1} \mod l \end{array} \right| .$$

(4) Compute $e := e + c \cdot kl$ for some c of bitsize $(1 - \alpha - 2\delta)n$.
(5) Compute $p = \frac{ed_p - 1}{k} - 1$ and $q = \frac{ed_q - 1}{l} - 1$. If either p or q is composite, repeat the whole algorithm.

OUTPUT: CRT-RSA-instance (e, N, d_p, d_q, p, q)

Notice that this key generation algorithm works as long as $\alpha + 2\delta \leq 1$. Namely, in Step 3 we compute a public key e of bitsize $(2\alpha + 2\delta - 1)n$, which is extended in Step 4 to bitsize αn. Therefore, we require that $\alpha \geq 2\alpha + 2\delta - 1$.

The above key generation is a slight variation of the GHM algorithm. In [10], the authors choose e, k, l first and afterwards compute d_p, d_q as inverses of e mod k, l, respectively. Then analogously to Step 4 above, they fill up d_q, d_q to the desired bitsize. Thus, their key generation requires that the sizes of d_p, d_q are at least the sizes of k, l. However, this condition is not fulfilled by a large portion of the RSA instances that we can attack. If the conditions of both key generations are fulfilled, one should however prefer the GHM method. It is more efficient, since one can generate p and q separately.

In the following experiments, we applied our key generation algorithm for varying sizes of e and d_p, d_q. The LLL reduction was carried out using a C-implementation of the provable L^2 reduction algorithm due to Nguyen and Stehlé [17]. The timings were performed on a 1GHz PC running Cygwin.

7.1 Experiments for Small e

All experiments in this section were done for 1000-bit N. For every fixed e, we looked for the maximal bitsize for d_p, d_q that gave us enough small vectors for recovering the secrets. In our experiments, we fixed the attack parameter $m = 2$ and tried different values of t.

In the table below, the third column provides the bound of Bleichenbacher-May which can be achieved using a 3-dimensional lattice. The fourth column provides the bound for an attack of Galbraith, Heneghan, and McKee [10], which is closely related to the attack described in this paper (see Appendix B for details on this GHM-attack). The δ-column gives the theoretical upper bound for the chosen parameters m, t and e. The 'asymp'-column gives the asymptotic bound which is reached when the lattice dimension goes to infinity.

e	d_p, d_q	BM[2]	GHM[10]	δ	asymp	lattice parameters	LLL
250 bit	332 bit	0.250	0.333	0.227	0.287	$m = 2, t = 0, \dim = 27$	2 sec
300 bit	299 bit	0.250	0.300	0.209	0.271	$m = 2, t = 0, \dim = 27$	2 sec
400 bit	239 bit	0.240	0.233	0.173	0.243	$m = 2, t = 0, \dim = 27$	2 sec
500 bit	199 bit	0.200	0.167	0.136	0.214	$m = 2, t = 0, \dim = 27$	2 sec
577 bit	168 bit	0.169	0.115	0.108	0.192	$m = 2, t = 0, \dim = 27$	2 sec
700 bit	119 bit	0.120	0.033	0.064	0.157	$m = 2, t = 0, \dim = 27$	2 sec
800 bit	79 bit	0.080	−0.033	0.027	0.128	$m = 2, t = 0, \dim = 27$	2 sec
900 bit	38 bit	0.040	−0.100	−0.009	0.100	$m = 2, t = 0, \dim = 27$	2 sec
900 bit	40 bit	0.040	−0.100	0.013	0.100	$m = 2, t = 1, \dim = 56$	93 sec
925 bit	29 bit	0.030	−0.117	−0.018	0.093	$m = 2, t = 0, \dim = 27$	2 sec
925 bit	31 bit	0.030	−0.117	0.006	0.093	$m = 2, t = 1, \dim = 56$	87 sec
950 bit	19 bit	0.020	−0.133	−0.027	0.087	$m = 2, t = 0, \dim = 27$	2 sec
950 bit	23 bit	0.020	−0.133	−0.001	0.087	$m = 2, t = 1, \dim = 56$	80 sec

In all the above experiments, we were able to recover the factorization of N. Experimentally, we see that our attack is much better than theoretically predicted. The reason is that for these RSA parameter settings, the shortest vectors are linear combinations of certain subsets of the lattice basis. I.e., the shortest vectors belong to some sublattice and the determinant calculation of the full lattice in Section 4 does not accurately capture the optimal choice of basis vectors. However, to identify the optimal sublattice structure for every fixed size e seems to be a difficult task.

Let us first comment on the results for 250-bit and 300-bit e. As can be seen in Appendix B, there exists an attack by Galbraith, Heneghan, and McKee [10] that is closely related to our new attack. Basically, they use a Coppersmith method for finding modular roots, to find the small root (k, l) of a polynomial f_e modulo e. The polynomial f_e is exactly our polynomial f taken modulo e. Since for $\alpha = 0.25, \alpha = 0.3$, the bound of the GHM-attack is superior to our new attack bound, the GHM-attack should be used for these cases instead of the new attack. However, if one uses the new attack, the lattice reduction algorithm chooses certain sublattices that still lead to the GHM-bound. This explains for these small values of α, why the experimental results are better than expected.

These were the only instances that we discovered, where Assumption 1 failed. Since the reduced basis vectors corresponded to the underlying structure of the GHM-attack, we were not able to eliminate three variables. However, we always found a polynomial of the form $(k+l-1)x_3x_4 - kl(x_3+x_4-1)$ in the Gröbner Basis, which directly yields k and l. The knowledge of k is sufficient to factor N in polynomial time, provided that e is large enough: Notice that

$$p = 1 - k^{-1} \mod e.$$

From a theorem of Coppersmith for factoring with high bits known [5], it follows that we can find p in polynomial time whenever $e \geq N^{\frac{1}{4}}$, which is satisfied in our experiments. We also made attacks for the case $e < N^{\frac{1}{4}}$, where we still got the secrets k, l. However, this information seems to be not sufficient for factoring N efficiently. This is consistent with the GHM-attack, where Galbraith, Heneghan, and McKee state that the attack only succeeds if the factorization of N can be extracted in polynomial time from the knowledge of the exposed k, l.

For $\alpha \geq 2/5$, i.e. e of bitsize at least 400, Assumption 1 was always valid. In all experiments, the Gröbner Basis of all polynomials yields the secret parameters (d_p, d_q, k, l) and therefore the factorization of N. The roots were found by using the F4 Gröbner Basis algorithm implemented in Magma V2.11-14. We would like to note that, when we did not include all candidates $f_0, \ldots, f_\ell$ but used only a few, it sometimes happened that we could eliminate two variables only. In that case, we were still able to retrieve the secrets, since the Gröbner Basis, where x_2 and x_4 were eliminated, then contained a polynomial with the terms $(d_p + (k-1)x_1 - d_px_3)$ and $(d_q + (l-1)x_1 - d_qx_3)$ in its factorization.

For e of bitsizes 400 up to 800, we actually rediscovered the bound $\frac{2}{5}(1-\alpha)$ by Bleichenbacher, May experimentally. Again the lattice reduction algorithm chose certain sublattices which in this case lead to the BM-bound. Even a moderate increasement of the lattice dimension did not give us any improvement in this range of e. Although our asymptotical bound always beats the BM-bound, we are not able to see this effect for small e, since going beyond the BM-bound requires high-dimensional lattice bases.

For e larger than 900 bits we can for the first time see the effect of increasing the lattice dimension and we are able to go slightly beyond the BM-bound. This effect intensifies for full size e, where the BM-bound does not give any results at all.

7.2 Experiments for Full Size e

Here we describe the experiments for RSA with a standard key generation for small CRT-exponents, which usually yields full size e. Namely, the parameters d_p, d_q are chosen for a fixed bitsize and e is the unique integer modulo $\phi(N)$ which is the inverse of d_p, d_q modulo $p-1$ and $q-1$, respectively.

N	d_p, d_q	δ	lattice parameters	LLL-time
1000 bit	10 bit	-0.015	$m = 2, t = 1, \dim = 56$	61 sec
1000 bit	13 bit	-0.002	$m = 2, t = 2, \dim = 95$	1129 sec
1000 bit	15 bit	0.002	$m = 3, t = 1, \dim = 115$	13787 sec
2000 bit	20 bit	-0.015	$m = 2, t = 1, \dim = 56$	255 sec
2000 bit	22 bit	-0.002	$m = 2, t = 2, \dim = 95$	1432 sec
2000 bit	32 bit	0.002	$m = 3, t = 1, \dim = 115$	36652 sec
5000 bit	52 bit	-0.015	$m = 2, t = 1, \dim = 56$	1510 sec
5000 bit	70 bit	-0.002	$m = 2, t = 2, \dim = 95$	18032 sec
10000 bit	105 bit	-0.015	$m = 2, t = 1, \dim = 56$	3826 sec
10000 bit	140 bit	-0.002	$m = 2, t = 2, \dim = 95$	57606 sec

Every experiment gave us sufficiently many polynomials with the desired roots over the integers, such that we could recover the factorization. The Gröbner computation never took more than 100 seconds and consumed a maximum of 300 MB.

Notice that for 10000-bit N, we can recover d_p, d_q of bitsize 140, which would not be possible by a square-root attack.

As in the experiments before, the δ-bound is very inaccurate. For lattice dimensions 56 and 95, we should not obtain any results at all, while experimentally we succeeded for d with bitsizes roughly a 0.010-fraction respectively a 0.013-fraction of N. On the other hand, our asymptotical bound states that we could in theory go up to a 0.073-fraction. Unfortunately, we are a tad bit away from the theoretical bound, since currently the best LLL-reductions only allow to reduce lattice bases of moderate size, when the base matrices have large entries. Let us give a numerical example. Theoretically, for $m = 10$ we find an optimal value of $t = 6$ which yields a bound of 0.063. However, this parameter choice results in a lattice dimension of 4200 which is clearly out of practical reach.

Our result guarantees that one can find the factorization of N for a sufficiently large – but fixed – lattice dimension for CRT-exponents d_p, d_q up to a 0.073-fraction. Moreover, it does not rule out that one can go beyond this bound. Even with our approach, the experimental results seem to indicate that an analysis of sublattice structures could lead to a better theoretical bound. We hope that these open problems stimulate further research in the exciting areas of lattice-based cryptanalysis and fast practical lattice reduction algorithms.

Acknowledgements:
We thank Antoine Joux and Ralph-Philipp Weinmann for discussions about Gröbner Bases, Maike Ritzenhofen for doing the Gröbner Basis computations in Magma, and Benne de Weger for his helpful comments.

References

1. A. Bauer, A. Joux, "Toward a Rigorous Variation of Coppersmith's Algorithm on Three Variables", Proceedings of Eurocrypt'07, LNCS 4515 [2007], pp. 361–378.
2. D. Bleichenbacher, A. May, "New Attacks on RSA with Small Secret CRT-Exponents", Proceedings of PKC 2006, LNCS 3958 [2006], pp. 1–13.
3. D. Boneh, "Twenty Years of Attacks on the RSA Cryptosystem", Notices of the American Mathematical Society 46 [1999], pp. 203-213.
4. D. Boneh, G. Durfee, "Cryptanalysis of RSA with Private Key d Less Than $N^{0.292}$ ", IEEE Transactions on Information Theory 46 [2000], 1339–1349.
5. D. Coppersmith, "Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities", Journal of Cryptology 10 [1997], pp. 233–260.
6. J.-S. Coron, "Finding Small Roots of Bivariate Integer Equations Revisited", Proceedings of Eurocrypt'04, LNCS 3027 [2004], pp. 492–505.
7. J.-S. Coron, "Finding Small Roots of Bivariate Integer Polynomial Equations: a Direct Approach", in these proceedings (Crypto'07).
8. D. Cox, J. Little, D. O'Shea, "Ideals, Varieties, and Algorithms", Second Edition, Springer-Verlag [1998].
9. ECRYPT - Hardness of the Main Computational Problems Used in Cryptography, IST-2002-507932, available at http://www.ecrypt.eu.org/documents/D.AZTEC.4-1.1.pdf
10. S. Galbraith, C. Heneghan, J. McKee, "Tunable Balancing of RSA", Proceedings of ACISP 2005, LNCS 3574 [2005], pp. 280-292.
11. S. Galbraith, C. Heneghan, J. McKee, "Tunable Balancing of RSA", full version of [10], http://www.isg.rhul.ac.uk/∼sdg/full-tunable-rsa.pdf.
12. N. Howgrave-Graham, "Finding Small Roots of Univariate Modular Equations Revisited", Cryptography and Coding, LNCS 1355 [1997], pp. 131–142.
13. E. Jochemsz, A. May, "A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants", Proceedings of Asiacrypt'06, LNCS 4284 [2006], pp. 267–282.
14. C.S. Jutla, "On Finding Small Solutions of Modular Multivariate Polynomial Equations", Proceedings of Eurocrypt'98, LNCS 1403 [1998], pp. 158–170.
15. A. Lenstra, H. Lenstra, Jr., L. Lovász, "Factoring Polynomials with Rational Coefficients", Mathematische Ann. 261 [1982], pp. 513–534.
16. A. May, "Cryptanalysis of Unbalanced RSA with Small CRT-Exponent", Proceedings of Crypto'02, LNCS 2442 [2002], pp. 242–256.
17. P. Nguyen, D. Stehlé, "Floating-Point LLL Revisited", Proceedings of Eurocrypt'06, LNCS 3494 [2006], pp. 215–233.
18. J.J. Quisquater, C. Couvreur, "Fast decipherment algorithm for RSA public-key cryptosystems" Electronic Letters 18 [1982], pp.905-907.
19. STORK - Strategic Roadmap for Crypto, IST-2002-38273, available at http://www.stork.eu.org/documents/RUB-D6-2_1.pdf
20. H.-M. Sun, M.-E. Wu, "An Approach Towards RSA-CRT with Short Public Exponent", IACR eprint, http://eprint.iacr.org/2005/053
21. H.-M. Sun, M.J. Hinek, M.-E. Wu, "On the Design of Rebalanced RSA-CRT", revised version of [20], http://www.cacr.math.uwaterloo.ca/techreports/2005/cacr2005-35.pdf
22. M. Wiener, "Cryptanalysis of Short RSA Secret Exponents", IEEE Transactions on Information Theory 36 [1990], pp. 553–558.

A Calculating the Bound and Finding More Polynomials

In this appendix, we show that the bound $\det(L)^{\frac{1}{d}} > 1$ of the implementation of our attack using Coppersmith's original method (Section 5) is equivalent to the bound (1) corresponding to an implementation following Coron's method (as used in Section 2). Moreover, we use a result from Jutla [14] to show that the vectors $\mathbf{v}'(x_1, x_2, x_3, x_4)_{d-\ell}$, $\ell \geq 2$, yield a list of at least three polynomials $f_0, \ldots, f_\ell$ having the same root (d_p, d_q, k, l).

One can check that

$$\det(L)^{\frac{1}{d}} = \det(B_1)^{\frac{1}{d}} = \left(\prod_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S'} (X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4})^{-1} \right)^{\frac{1}{d}} \cdot (e^2)^{\frac{s}{d}}.$$

So the bound $\det(L)^{\frac{1}{d}} > 1$ implies that

$$\left(\prod_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S'} (X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4}) \right) < (e^2)^s. \tag{2}$$

Let us substitute e^2 by $\frac{W}{X_1 X_2}$. We observe that the difference between the monomials of $M \setminus S'$ and $M \setminus S$ is s times the monomial $x_1 x_2$. Multiplying both sides by $(X_1 X_2)^s$ yields

$$X_1^{s_1} X_2^{s_2} X_3^{s_3} X_4^{s_4} < W^s, \text{ for } s_j = \sum_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S} i_j \quad \text{and } s = \sum_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S} 1.$$

Notice that this condition is equivalent to the condition (1) given in Section 2.

It follows that if this bound holds, then applying Coppersmith's method gives us a polynomial $f_0(x_1, x_2, x_3, x_4)$ from the coefficient vector $\mathbf{v}'(x_1, x_2, x_3, x_4)_d$, such that f_0 has the desired root (d_p, d_q, k, l) over the integers. But in order to extract the root, we have to construct at least two more polynomials which share the same root.

We will prove now that it is always possible to construct any constant number of polynomials with the same common root provided that condition (1) is satisfied, at the cost of a slightly larger error term ϵ in the construction. Therefore, we use a theorem of Jutla [14], which gives us a lower bound for the length of any Gram-Schmidt vector in an LLL-reduced basis. Namely,

$$\|\mathbf{b}_i^*\| \geq 2^{\frac{-(i-1)}{4}} \left(\frac{\det(L)}{b_{\max}^{m-i}} \right)^{\frac{1}{i}} \qquad \text{for } i = 1 \ldots d,$$

where $b_{\max}$ is the maximal length of the Gram-Schmidt orthogonalization of the matrix A (the matrix before starting the LLL-reduction process). Following the analysis of [14], it can be checked that in our attack, $b_{\max} = e^2$. Therefore,

$\|\mathbf{b}_i^*\| > 1$ reduces to

$$2^{\frac{-(i-1)}{4}} \left(\frac{\left(\prod_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S'} (X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4})^{-1}\right) \cdot (e^2)^s}{(e^2)^{d-i}} \right)^{\frac{1}{i}} > 1.$$

Since $2^{\frac{-(i-1)}{4}}$ does not depend on N, we let it contribute to an error term ϵ. This simplifies our condition to

$$\prod_{x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in M \setminus S'} (X_1^{i_1} X_2^{i_2} X_3^{i_3} X_4^{i_4}) < (e^2)^{s-(d-i)},$$

Notice that for $i = d$, we obtain the same bound as in (2). In Section 4, we have seen that $s = m^4(1 + o(1))$. So as long as $d - i = o(m^4)$, the asymptotic bound does not change and we get just another error term that contributes to ϵ. This is clearly satisfied if $d - i = \ell$ for some constant ℓ. Thus, all polynomials $f_0, \ldots, f_\ell$ corresponding to the coefficient vectors $\mathbf{v}'(x_1, x_2, x_3, x_4)_{d-i}$, $i = 0 \ldots \ell$, share the common root (d_p, d_q, k, l), as desired.

B A Related Attack by Galbraith, Heneghan, and McKee

In Section 7.1 we noted that for very small e, there is an attack by Galbraith, Heneghan, and McKee [10, Section 5.1] that works better than our new attack. In this appendix, we briefly describe this GHM-attack and its relation to our new attack.

Recall that for our new attack, we multiply the equations

$$ed_p + k - 1 = kp \quad \text{and} \quad ed_q + l - 1 = lq$$

to obtain the polynomial

$$f(x_1, x_2, x_3, x_4) = e^2 x_1 x_2 + e x_1 x_4 - e x_1 + e x_2 x_3 - e x_2 - (N-1) x_3 x_4 - x_3 - x_4 + 1$$

with the small root (d_p, d_q, k, l).

In their attack in [10, Section 5.1], Galbraith, Heneghan, and McKee do essentially the same, but modulo e. Hence, the goal of their attack is to find the modular root (k, l) of the polynomial $f_e(x_3, x_4) = (N - 1)x_3 x_4 + x_3 + x_4 - 1$ modulo e. This polynomial f_e, with monomials $1, x_3, x_4, x_3 x_4$ has a well-known [5] bound

$$X_3 X_4 < e^{\frac{2}{3}}.$$

that specifies for which upper bounds X_3, X_4 of x_3, x_4 the root can be found in polynomial time. Substituting $X_3 = X_4 = N^{\alpha+\delta-\frac{1}{2}}$, and $e = N^\alpha$, we find the attack bound

$$\delta < \frac{1}{2} - \frac{2}{3}\alpha.$$

For very small α (for instance $\alpha = 0.25$ and $\delta = 0.3$), this bound is superior to the bound obtained by our new attack, and for these cases, the GHM-attack should be preferred to the new attack.