Solving Systems of Modular Equations in One Variable: How Many RSA-Encrypted Messages Does Eve Need to Know? *

Alexander May, Maike Ritzenhofen

Faculty of Mathematics
Ruhr-Universität Bochum, 44780 Bochum, Germany
alex.may@ruhr-uni-bochum.de
maike.ritzenhofen@ruhr-uni-bochum.de

Abstract. We address the problem of polynomial time solving univariate modular equations with mutually co-prime moduli. For a given system of equations we determine up to which size the common roots can be calculated efficiently. We further determine the minimum number of equations which suffice for a recovery of all common roots. The result that we obtain is superior to Håstad's original RSA broadcast attack, even if Håstad's method is combined with the best known lattice technique due to Coppersmith. Namely, our reduction uses a slightly different transformation from polynomial systems to a single polynomial. Thus, our improvement is achieved by optimal polynomial modelling rather than improved lattice techniques. Moreover, we show by a counting argument that our results cannot be improved in general. A typical application for our algorithm is an improved attack on RSA with a smaller number of polynomially related messages.

Key words: Chinese Remaindering, Coppersmith's method, Håstad's attack, systems of univariate modular polynomials

1 Introduction

The RSA cryptosystem [14] is the public key cryptosystem which is most widely used in practice. Therefore, it has attracted the interest of many cryptanalysts since its invention in 1977 (compare e. g. [2]). In the following, let us denote by $N = pq$ the RSA modulus with prime factors p and q, and let $\mathbb{Z}_N$ denote the ring of integers modulo N. Let e be the public exponent, and let $d = e^{-1} \pmod{\varphi(N)}$ be the private key.

Attacks on RSA intend either to factorize the modulus and thereby recover the private key, or to compute e-th roots modulo N, i. e. to decrypt ciphertexts.

* This research was supported by the German Research Foundation (DFG) as part of the project MA 2536/3-1

The equivalence or inequivalence of these two problems is still open. However, partial results are known in restricted models [3, 4, 10].

In this paper we deal with the problem of extracting e-th roots. This is the well-known RSA problem: Given an RSA modulus N, a public exponent e and a ciphertext $c \equiv m^e \pmod{N}$, find the corresponding plaintext m.

If $m^e < N$, the equation does not only hold in $\mathbb{Z}_N$ but over the integers, and we can calculate m easily. This implies that encrypting small messages with small public exponents is insecure.

Let us look at the inhomogeneous case. Namely, suppose the most significant bits are known so that the unknown part remains small enough. Then we get the equation $(\tilde{m} + x)^e \equiv c \pmod{N}$, with $\tilde{m}$ denoting the known, x the unknown part of the message. D. Coppersmith [6] showed that this inhomogeneous case can be solved efficiently under the same condition $x^e < N$.

Precisely, he showed that given a composite integer N and a univariate polynomial $f(x) \in \mathbb{Z}_N[x]$ of degree δ one can determine all zeros smaller than $N^{\frac{1}{\delta}}$ efficiently. Hence, $(\tilde{m} + x)^e \equiv c \pmod{N}$ can be solved if $|x| < N^{\frac{1}{e}}$.

Now we may ask what happens if we get further information in form of additional polynomials? Can we then determine larger zeros as well?

There are two variants of systems of polynomial modular equations. Either there exist equations with the same modulus or all moduli are different. The first case was considered in Coppersmith, Franklin, Patarin and Reiter [7]. They showed that it is usually sufficient to have two equations $f_1(x) \equiv 0 \pmod{N}$ and $f_2(x) \equiv 0 \pmod{N}$ in order to recover the common roots. Let a be the common solution of the two equations. Then, $f_1(x)$ and $f_2(x)$ share a factor $(x - a)$. Computing the greatest common divisor $\gcd(f_1(x), f_2(x)) \pmod{N}$ reveals this factor if it is the only common factor. In the rare cases where the greatest common divisor is not linear, the method fails and further polynomials are needed. The running time of this method is $O(\delta \log^2 \delta)$ where δ is the degree of the given polynomials.

It is worth pointing out that a scenario with two RSA encryptions under coprime public exponents (e_1, e_2) and a common modulus N is a special case of this setting. Namely, an attacker has to find the common root m of $f_1(x) = x^{e_1} - m^{e_1} \pmod{N}$ and $f_2(x) = x^{e_2} - m^{e_2} \pmod{N}$. G. Simmons [16] has presented a neat attack for this special setting with running time polynomial in the bitlength of (e_1, e_2). Namely, one computes integers u_1, u_2 such that $u_1 e_1 + u_2 e_2 = 1$ with the help of the Extended Euclidean Algorithm. This gives us $m = (m^{e_1})^{u_1} (m^{e_2})^{u_2} \pmod{N}$.

In this work, we focus on equations with different moduli $N_1, N_2, \ldots, N_k \in \mathbb{N}$. Without loss of generality, we assume that all moduli are composite as modular equations over finite fields can be solved efficiently (compare e. g. [1], Chapter 7.4). We further assume that the N_i, $i = 1, \ldots, k$, are relatively prime. In case of our main application, RSA-moduli, we can otherwise compute prime factors of the N_i by computing the greatest common divisor.

Before we define our polynomial roots problem in general, let us give a motivating cryptographic application. This application was introduced by J. Håstad in [8, 9] and can be considered as an analogue of Simmon's attack in the setting of different RSA moduli. A user wishes to send the same message m to several participants having different moduli and using plain RSA encryption without padding techniques. Suppose these users share the same public exponent $e = 3$. Then, an attacker obtains three equations $m^3 \equiv c_i \pmod{N_i}$ for $i = 1, 2, 3$. He can make use of the fact that the N_i are relatively prime and combine the equations by the Chinese Remainder Theorem. Thus, he gets $m^3 \pmod{N_1 N_2 N_3}$ and is able to determine m in $\mathbb{Z}$ as $m^3 < N_1 N_2 N_3$. Therefore, the attacker solves the system of polynomial equations $f_i(x) \equiv x^3 - c_i \equiv 0 \pmod{N_i}$, $i = 1, 2, 3$, with the common root m.

Now let us generalize to arbitrary polynomial equations. We define the problem of solving *systems of modular univariate polynomial equations (SMUPE-problem)*.

Definition 1 (SMUPE-problem). *Let $k \in \mathbb{N}$, $\delta_1, \ldots, \delta_k \in \mathbb{N}$, and let $N_1, \ldots, N_k \in \mathbb{N}$ be mutually co-prime composite numbers of unknown factorization. Suppose $N_1 < N_2 < \ldots < N_k$. Let $f_1(x), \ldots, f_k(x)$ be polynomials of degree $\delta_1, \ldots, \delta_k$ in $\mathbb{Z}_{N_1}[x], \ldots, \mathbb{Z}_{N_k}[x]$, respectively. Let*

$$\begin{aligned} f_1(x) &\equiv 0 \pmod{N_1} \\ f_2(x) &\equiv 0 \pmod{N_2} \\ &\vdots \\ f_k(x) &\equiv 0 \pmod{N_k} \end{aligned} \tag{1}$$

be a system of univariate polynomial equations.

Let $X \leq N_1$, $X \in \mathbb{R}$. Find all common roots x_0 of (1) with size $|x_0| < X$.

Our goal is to compute an upper bound X for which the SMUPE-problem is solvable in time polynomial in $\prod_{i=1}^{k} \delta_i$ and in the bitlength of $\prod_{i=1}^{k} N_i$. This upper bound will give us a condition on the number of equations k in terms of δ_i and N_i. This will enable us to compute the minimal k such that the SMUPE-problem can be computed up to the bound $X = N_1$, i.e. system (1) can be solved efficiently.

J. Håstad [9] gave the following algorithm for solving the SMUPE-problem. Let $\delta \in \mathbb{N}$ be the maximum degree of all polynomials occuring in the system, i. e. $\delta := \max_{i=1,\ldots,k}\{\delta_i\}$. One first multiplies the given polynomials with $x^{\delta - \delta_i}$ to adjust their degrees. Then one combines the resulting polynomials using the Chinese Reminder Theorem to a univariate polynomial $f(x)$ with the same roots modulo $\prod_{i=1}^{k} N_i$. Applying lattice reduction methods, J. Håstad derived $k > \frac{\delta(\delta+1)}{2}$ as a lower bound on the number of polynomials for efficiently finding all roots x_0 with $|x_0| < N_1$. This bound can be easily improved to $k \geq \delta$ by directly applying Coppersmith's lattice techniques [6] to $f(x)$ (see e.g. [2]).

Our contribution: We give a different construction to combine all k polynomial equations into a single equation $f(x) \equiv 0 \pmod{\prod_{i=1}^{k} N_i}$. Instead of multiplying the polynomials by powers of x like in Håstad's approach, we take powers of the polynomials $f_i(x)$ themselves. This results in the condition $\sum_{i=1}^{k} \frac{1}{\delta_i} \geq 1$ for solving the SMUPE-problem for all x_0 with $|x_0| < N_1$. In case all polynomials share the same degree δ this corresponds to Håstad's condition $k \geq \delta$. For polynomials of different degrees, however, our new condition is superior. Especially, a few polynomials of low degree suffice.

The paper is organized as follows. In Section 2, we review Coppersmith's result from [6] and the Chinese Remainder Theorem for polynomials. In Section 3, we prove the new sufficient condition on the number of polynomials that is needed to recover all common roots efficiently. The improved RSA broadcast attack is given as an application in Section 4. In Section 5, we show that our condition cannot be improved in general by giving an example for which the condition is optimal.

2 Preliminaries

The problem of solving modular univariate polynomial equations is believed to be difficult in general. Under some restrictions on the roots however, this is not the case. In [6], D. Coppersmith showed how to provably determine zeros of modular univariate equations with sufficiently small size.

Theorem 1 (Coppersmith [6]). *Let $f(x)$ be a monic polynomial of degree $\delta \in \mathbb{N}$ in one variable modulo an integer N of unknown factorization. Let X be a bound on the desired solution x_0. If $X \leq N^{\frac{1}{\delta}}$ then we can find all integers x_0 such that $f(x_0) \equiv 0 \pmod N$ and $|x_0| \leq X$ in time $O(\delta^5(\delta + \log N)\log N)$.*

The running time can be achieved by using an algorithm of Nguyen, Stehlé [13] for the LLL lattice basis reduction step (see [11, 12]).

The SMUPE-problem can be reduced to the problem of solving a single univariate polynomial equation by combining the equations into a single one with the same solutions. Then we can apply Theorem 1. A possible way to combine equations is by Chinese Remaindering which is described e. g. in [9, 15].

Theorem 2 (Chinese Remainder Theorem). *Let $k \in \mathbb{Z}$. Let $\delta \in \mathbb{N}$, $\delta > 1$. For $i = 1, \ldots, k$ let $N_i \in \mathbb{N}$ be pairwise relatively prime numbers, and let $f_i(x) \in \mathbb{Z}[x]$ be polynomials of degree δ.*
Then there exists a unique polynomial $f(x)$ modulo $M := \prod_{i=1}^{k} N_i$ such that

$$f(x) \equiv f_i(x) \pmod{N_i} \qquad (2)$$

The polynomial $f(x)$ can be determined in time $O(\delta \log^2 M)$.

Proof. Let $M := \prod_{i=1}^{k} N_i$, $M_i := \frac{M}{N_i}$ and M_i' be the inverse of M_i modulo N_i for $i = 1, \ldots, k$. The existence of such an inverse is guaranteed by $\gcd(M_i, N_i) = 1$. Then

$$f(x) = \sum_{i=1}^{k} M_i M_i' f_i(x)$$

is the desired solution. If we look at $f(x)$ modulo N_j for $j \in \{1, \dots, k\}$, all summands with index $i \neq j$ cancel out (as N_j divides M_i) and $M_j M'_j f_j(x) \equiv f_j(x) \pmod{N_j}$.

Now suppose that $g(x)$ is another solution fulfilling the required conditions. Then, $f(x) - g(x) \equiv 0 \pmod{N_i}$ for all $i = 1, \dots, k$, and therefore also $f(x) \equiv g(x) \pmod{M}$.
Multiplication modulo M and calculating the inverses by the Extended Euclidean Algorithm can be performed in time $O(\log^2 M)$. Determining all coefficients of f then gives us $O(\delta \log^2 M)$ for the complete algorithm. □

3 An Improved Algorithm for Solving SMUPE

For notational convenience let us briefly recall the SMUPE-problem. Given $k \in \mathbb{N}$, $N_1, \dots, N_k \in \mathbb{N}$, mutually co-prime composite numbers of unknown factorization, such that $N_1 < \ldots < N_k$, and a system of polynomial equations

$$\begin{aligned} f_1(x) &\equiv 0 \pmod{N_1} \\ f_2(x) &\equiv 0 \pmod{N_2} \\ &\vdots \\ f_k(x) &\equiv 0 \pmod{N_k}, \end{aligned} \tag{1}$$

where $f_1(x), \dots, f_k(x)$ are of degree $\delta_1, \dots, \delta_k \in \mathbb{N}$ in $\mathbb{Z}_{N_1}[x], \dots, \mathbb{Z}_{N_k}[x]$, respectively.
Let $X \leq N_1$, $X \in \mathbb{R}$. Recover all solutions x_0 of (1) with $|x_0| < X$.

Considering for example Coppersmith's method (Theorem 1) for the first equation in (1), only small roots x_0 with $|x_0| < N_1^{\frac{1}{\delta_1}}$ can be found in polynomial time. By considering further equations this bound can be improved until all solutions can be found eventually.

By Håstad's algorithm in combination with Theorem 1 the condition $k \geq \delta$ with $\delta := \max_{i=1,\dots,k}\{\delta_i\}$ is sufficient to solve a system of equations efficiently. However, this condition is clearly not optimal as the following trivial example shows. Let $N_1 < \ldots < N_4$ and take the following equations.

$$\begin{aligned} x^3 &\equiv c_1 \pmod{N_1} \\ x^3 &\equiv c_2 \pmod{N_2} \\ x^3 &\equiv c_3 \pmod{N_3} \\ x^5 &\equiv c_4 \pmod{N_4} \end{aligned}$$

Then $k = 4 < 5 = \delta$, i.e. the condition is not fulfilled. However, if we just take the first three equations, we are able to compute all common solutions smaller than N_1. This gives us the intuition that the proportion of higher and lower degrees of the polynomials ought to be taken into account. Let us now change the given example a little bit into a non-trivial one, so that no subsystem of the equations fulfills the sufficient condition.

$$\begin{aligned} x^3 &\equiv c_1 \pmod{N_1} \\ x^3 &\equiv c_2 \pmod{N_2} \\ x^5 &\equiv c_3 \pmod{N_3} \\ x^5 &\equiv c_4 \pmod{N_4} \end{aligned}$$

The parameters k and δ and the N_i remain the same. Can we still determine all solutions? We notice that we can transform the first equation by squaring into

$$x^6 \equiv 2c_1x^3 - c_1^2 \pmod{N_1^2}.$$

Applying Theorem 1 to this equation, we can find all solutions x for which $|x| < (N_1^2)^{\frac{1}{6}} = N_1^{\frac{1}{3}}$ holds. This is the same bound which we get for the roots of the original equation $x^3 \equiv c_1 \pmod{N_1}$. We proceed with the second equation in the same way, then multiply the two other equations by x and finally combine all the equations by Theorem 2 (Chinese Remainder Theorem). This gives us

$$x^6 \equiv a_1(2c_1x^3 - c_1^2) + a_2(2c_2x^3 - c_2^2) + a_3xc_3 + a_4xc_4 \pmod{N_1^2N_2^2N_3N_4},$$

where the a_i are the coefficients from the Chinese Remainder Theorem, i.e. $a_i \equiv 1 \pmod{N_i}$, $a_i \equiv 0 \pmod{N_j}$, $j \neq i$. The above equation can be solved in $\mathbb{Z}$ for x with $|x| < (N_1^2N_2^2N_3N_4)^{\frac{1}{6}}$. This condition is fulfilled for any x with $|x| < N_1 = (N_1^6)^{\frac{1}{6}} \leq (N_1^2N_2^2N_3N_4)^{\frac{1}{6}}$. Therefore, we can determine all solutions of the above system of equations, although the condition $k \geq \delta$ is not fulfilled.

In order to generalize our approach we make the following crucial observation. Let $f(x)$ be a polynomial of degree δ. Let $f(x) \equiv 0 \pmod{N}$ for $N \in \mathbb{N}$, and let $m \in \mathbb{N}$. Then $g(x) := f^m(x) \equiv 0 \pmod{N^m}$. The solutions x with $|x| < N$ of the two equations remain unchanged. Moreover, with Coppersmith's Theorem 1 we can determine those solutions for which the condition $|x| < N^{\frac{1}{\delta}} \Leftrightarrow |x| < (N^m)^{\frac{1}{m\delta}}$ holds. Thus, Coppersmith's bound is invariant under taking powers of the polynomial $f(x)$.

As opposed to our approach, in Håstad's algorithm one does not take powers of the polynomials but multiplications of polynomials with powers of x. This increases the degree of the polynomial but leaves the modulus unchanged. Let $f(x)$ be a polynomial of degree δ with $f(x) \equiv 0 \pmod{N}$ for $N \in \mathbb{N}$. Then with $\gamma > \delta$ the equation $g(x) := x^{\gamma-\delta}f(x) \equiv 0 \pmod{N}$ contains all the solutions x of $f(x)$ with $|x| < N$. However, applying Coppersmith's method to determine roots of $g(x)$ we only get roots x with $|x| < N^{\frac{1}{\gamma}} < N^{\frac{1}{\delta}}$. So obviously, Coppersmith's bound is not invariant under multiplication with powers of x. This explains why we obtain a superior bound on the size of the roots.

In the following analysis we will restrict ourselves to monic polynomials. If one of the given polynomials $f_i(x)$ is not monic, either the coefficient of the leading monomial is invertible, or we can find a factor of the modulus. In the first case, we make the polynomial monic by multiplication with the inverse of the leading coefficient. In the latter case, we obtain for RSA moduli the complete factorization, which in turn allows for efficiently solving this polynomial equation modulo the prime factors.

Theorem 3. *Let $(f_i, \delta_i, N_i), i = 1, \ldots, k$, be an instance of the SMUPE-problem with monic f_i. Define $M := \prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}}$ with $\delta := lcm\{\delta_i, i = 1, \ldots, k\}$. Then the SMUPE-problem can be solved for all x_0 with*

$$|x_0| < M^{\frac{1}{\delta}}$$

in time $O(\delta^6 \log^2 M)$.

Proof. Let x_0 be a solution of the system of polynomial equations (1). Then x_0 is a solution of

$$f_i^{\frac{\delta}{\delta_i}}(x) \equiv 0 \pmod{N_i^{\frac{\delta}{\delta_i}}} \text{ for all } i = 1, \ldots, k.$$

All these equations have common degree δ and are monic.

Combining them by Chinese Remaindering yields a polynomial $f(x)$ of degree δ such that x_0 is a solution of $f(x) \equiv 0 \pmod{M}$ with $M := \prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}}$. Moreover, this polynomial is still monic.

For the coefficient a_δ of the monomial x^δ in $f(x)$ it holds that $a_\delta \equiv 1 \pmod{N_i^{\frac{\delta}{\delta_i}}}$ for all $i = 1, \ldots, k$ and therefore $a_\delta \equiv 1 \pmod{M}$.

The above step can be performed in time $O(\delta \log^2 M)$ by Theorem 2. With Theorem 1 all solutions x_0 of the above equation which fulfill $|x_0| \leq M^{\frac{1}{\delta}} = (\prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}})^{\frac{1}{\delta}}$ can be found in time $O(\delta^5(\delta + \log M) \log M)$. The result can therefore be obtained in time $O(\delta^6 \log^2 M)$. □

Remark 1. The same result is obtained by applying Coppersmith's method [6] directly to the polynomials $f_1(x), \ldots, f_k(x)$ instead of $f(x)$.

Theorem 3 immediately gives us a sufficient condition on k and the δ_i for solving the SMUPE-problem for all $x_0 \in \mathbb{Z}_{N_1}$.

Corollary 1. *The SMUPE-problem can be solved for all $x_0 \in \mathbb{Z}_{N_1}$ in time $O(\delta^6 \log^2 M)$ provided that*

$$\sum_{i=1}^{k} \frac{1}{\delta_i} \geq 1. \tag{3}$$

Proof. Let x_0 be a common solution to all the equations. An application of Theorem 3 gives us $|x_0| < M^{\frac{1}{\delta}} := (\prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}})^{\frac{1}{\delta}}$ as an upper bound for all roots that can be computed in time $O(\delta^6 \log^2 M)$. As $(\prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}})^{\frac{1}{\delta}} \geq \prod_{i=1}^{k} N_1^{\frac{1}{\delta_i}} = N_1^{\sum_{i=1}^{k} \frac{1}{\delta_i}} \geq N_1$ all solutions $x_0 \in \mathbb{Z}_{N_1}$ can be found. □

This gives us an algorithm to solve the SMUPE-problem with running time polynomial in the bitsize of the N_i, $i = 1, \ldots, k$, if δ is polynomial in the bitsize of the N_i.

Comparing this to the result due to Håstad and Coppersmith we observe that in the case $\delta := \delta_1 = \ldots = \delta_k$ the sufficient condition is $k \geq \delta$ with both methods. For different δ_i however, our method is always superior. Taking e.g. the illustrating example with public exponents $(3, 3, 5, 5)$ from the beginning of this section, we see that our new condition $\frac{1}{3} + \frac{1}{3} + \frac{1}{5} + \frac{1}{5} = \frac{16}{15} \geq 1$ is fulfilled.

4 Application: RSA with Polynomially Related Messages

A typical example in which polynomially related messages occur is an RSA broadcast scenario. Assume a user wants to broadcast a message m to k different users using an RSA encryption scheme with public exponents $e_1, \ldots, e_k$ and coprime public moduli $N_1 < \ldots < N_k$. From the ciphertexts $c_1 \pmod{N_1}, \ldots, c_k \pmod{N_k}$ an attacker can compute the message m if m is smaller than the upper bound given in Theorem 3. He sets $f_i(x) = x^{e_i} - c_i \pmod{N_i}$ and applies Theorem 3.

In order to avoid sending various encryptions of the same message, a user might add some randomness r_i and then encrypt the linearly related messages $(m + r_i)$, $i = 1, \ldots, k$, instead of m. However, if the attacker gets to know the randomness, he can calculate $F_i(x) := f_i(x + r_i) \pmod{N_i}$ and analyze the system of equations $F_i(x) \equiv 0 \pmod{N_i}$, $i = 1, \ldots, k$. As degree, modulus and leading coefficient are the same for $F_i(x)$ and $f_i(x)$, the upper bound on m, up to which m can be recovered efficiently, also remains unchanged. More generally, taking polynomially related messages instead of linearly related ones, the degree of $F_i(x)$, $i = 1, \ldots, k$, changes from e_i to $e_i\gamma_i$, where γ_i is the degree of the known polynomial relation.

Theorem 4. *Let $k \in \mathbb{N}$, (e_i, N_i), $i = 1, \ldots, k$, be RSA public keys with $N_1 < N_2 < \ldots < N_k$ and co-prime N_i. Furthermore, let $m \in \mathbb{Z}_{N_1}$ and let $g_i(x) \in \mathbb{Z}[x]$ be polynomials of degree $\gamma_i \in \mathbb{N}$ with $a_{i\gamma_i}$ the coefficient of x^{γ_i} for $i = 1, \ldots, k$. Let $c_1, \ldots, c_k$ be the RSA-encryptions of $g_i(m)$ under the public key (e_i, N_i). Define $\delta_i := e_i\gamma_i$ and $M := \prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i}}$ with $\delta := lcm\{\delta_i, i = 1, \ldots, k\}$.*
Then an adversary can recover the message m in time $O(\delta^6 \log^2 M)$ provided that

$$\sum_{i=1}^{k} \frac{1}{\delta_i} \geq 1.$$

Proof. Without loss of generality we assume that all $a_{i\gamma_i}$ are invertible modulo N_i. (Otherwise $\gcd(a_{i\gamma_i}, N_i)$ and $\frac{N_i}{\gcd(a_{i\gamma_i}, N_i)}$ will give us the factorization of N_i for at least one $i \in \{1, \ldots, k\}$. We can then compute m modulo the prime factors. This can be done efficiently (see [1])).

We are looking for a solution m of $f_i(x) := g_i(x)^{e_i} - c_i \equiv 0 \pmod{N_i}$, $i = 1, \ldots, k$. However, the polynomials $f_i(x)$ are not necessarily monic. Therefore, we modify them slightly to be able to apply Corollary 1. Let $F_i(x) := a_{i\gamma_i}^{-e_i}\big(g_i(x)^{e_i} - c_i\big) \pmod{N_i}$, $i = 1, \ldots, k$. Hence, $F_i(x)$ is a monic polynomial of degree $\delta_i = e_i\gamma_i$. The theorem then directly follows as an application of Corollary 1. □

5 Optimality of Our Bound for Solving SMUPE

In this section, we will see that the condition $|x_0| < M^{\frac{1}{\delta}}$ for efficiently solving the SMUPE-problem given in Theorem 3 is optimal if the moduli N_i are prime powers. This implies that the condition cannot be improved in general, unless we make use of the structure of the moduli or of the specific polynomials occuring in the system. Thus, our argument does not exclude the existence of superior conditions for special moduli, e.g. square-free N_i. Moreover, our formula captures the intuition that equations of low degree δ_i comprise more information since they contribute to the sum in (3) with a larger term $\frac{1}{\delta_i}$ than equations with higher degree.

The counting argument that we use is a generalization of the argument in [5] to systems of polynomial equations instead of a single equation.

Let $k \in \mathbb{N}$. Let $p_1, \ldots, p_k$ be different prime numbers, $\delta_1, \ldots, \delta_k \in \mathbb{N}$ and $N_1 := p_1^{\delta_1}, \ldots, N_k := p_k^{\delta_k}$. Suppose $N_1 < \ldots < N_k$. Let us look at the following system of polynomial equations.

$$\begin{aligned} f_1(x) &:= x^{\delta_1} \equiv 0 \pmod{N_1} \\ f_2(x) &:= x^{\delta_2} \equiv 0 \pmod{N_2} \\ &\vdots \\ f_k(x) &:= x^{\delta_k} \equiv 0 \pmod{N_k} \end{aligned} \tag{4}$$

We would like to determine all solutions x_0 of this system with $|x_0| < N_1 = p_1^{\delta_1}$. An application of Theorem 1 to a single equation $f_i(x) \equiv 0 \pmod{N_i}$ efficiently yields all solutions x_0 with $|x_0| < (N_i)^{\frac{1}{\delta_i}} = p_i$. Furthermore, each multiple of p_i is a solution of $f_i(x) \equiv 0 \pmod{N_i}$. Thus, if x_0 is a multiple of $\prod_{i=1}^k p_i$, then x_0 is a common zero of all the polynomials.

Let $\delta := \operatorname{lcm}\{\delta_i, i = 1, \ldots, k\}$. We apply the same method as in the proof of Theorem 3 to the polynomial equations in system (4). Namely, we take their $\frac{\delta}{\delta_i}$th powers and combine them by Chinese Remaindering (Theorem 2). This gives us an equation $f(x) \equiv x^{\delta} \pmod{M}$ with $M := \prod_{i=1}^k N_i^{\frac{\delta}{\delta_i}} = \prod_{i=1}^k p_i^{\delta}$ with the same roots as in (4).

We assume that $M^{\frac{1}{\delta}} < N_1$. Otherwise $M^{\frac{1}{\delta}} \geq N_1 > |x_0|$, i.e. the condition of Theorem 3 is fulfilled and there is nothing to be shown. Therefore, let $\epsilon > 0$ such that $M^{\frac{1}{\delta}+\epsilon} < N_1$. Suppose now we could calculate all simultaneous solutions x_0 of the system such that $|x_0| < M^{\frac{1}{\delta}+\epsilon} = (\prod_{i=1}^k p_i)^{1+\delta\epsilon}$. Since we know that every integer multiple of $\prod_{i=1}^k p_i$ is a root of (4), the number of roots is roughly

$2(\prod_{i=1}^{k} p_i)^{\delta\epsilon}$. This implies that we have exponentially many roots x_0 with $|x_0| < M^{\frac{1}{\delta}+\epsilon}$, which we cannot even output in polynomial time. Consequently, there is no polynomial time algorithm that improves upon the exponent in the condition $|x_0| < M^{\frac{1}{\delta}}$ of Theorem 3.

6 Acknowledgments

We thank the anonymous reviewers of PKC 2008 for their helpful comments.

References

1. E. Bach, J. Shallit, "Algorithmic Number Theory, Volume 1, efficient algorithms", The MIT Press, 1996
2. D. Boneh, "Twenty years of attacks on the RSA cryptosystem", Notices of the AMS, 1999.
3. D. Boneh, R. Venkatesan, "Breaking RSA May Not Be Equivalent To Factoring", Advances in Cryptology (Eurocrypt 1998), Lecture Notes in Computer Science, Volume 1233, pages 36-51, 2001
4. D. Brown, "Breaking RSA May Be As Difficult As Factoring", Cryptology ePrint Archive Report 2005/380, 2005
5. D. Coppersmith, "Finding Small Solutions to Small Degree Polynomials", CaLC 2001, Lecture Notes in Computer Science, Volume 2146, pages 20-31, 2001
6. D. Coppersmith, "Small solutions to polynomial equations and low exponent vulnerabilities", Journal of Cryptology, Volume 10(4), pp. 223-260, 1997
7. D. Coppersmith, M. Franklin, J. Patarin and M. Reiter, "Low-exponent RSA with related messages", Advances in Cryptology (Eurocrypt 1996), Lecture Notes in Computer Science, Volume 1070, pages 1-9, Springer Verlag, 1996.
8. J. Håstad, "On Using RSA with Low Exponent in a Public Key Network", In Advances in Cryptology (Crypto '85), Lecture Notes in Computer Science, Volume 218, pages 403-408, Springer Verlag, 1985
9. J. Håstad, "Solving Simultaneous Modular Equations of Low Degree", SIAM Journal on Computing, Vol. 17, no. 2, pp. 336–341, 1988
10. G. Leander, A. Rupp, "On the Equivalence of RSA and Factoring Regarding Generic Ring Algorithms", In Advances in Cryptology (Asiacrypt 2006), Lecture Notes in Computer Science, Volume 4284, pages 241-251, Springer Verlag, 2006
11. A. K. Lenstra, H. W. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients", Mathematische Annalen, Volume 261, pp. 513–534, 1982
12. A. May, "Using LLL-Reduction for Solving RSA and Factorization Problems: A Survey", LLL+25 Conference in honour of the 25th birthday of the LLL algorithm, 2007, online available at http://www.cits.rub.de/personen/may.html
13. P. Q. Nguyen, D. Stehlé, "Floating Point LLL Revisited", In Advances in Cryptology (Eurocrypt 2005), Lecture Notes in Computer Science, Volume 3494, pages 215 - 233, Springer-Verlag, 2005.
14. R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Volume 21(2), pp. 120-126, 1978
15. V. Shoup, "A Computational Introduction to Number Theory and Algebra", Cambridge University Press, 2005

16. G. Simmons, “A “Weak” Privacy Protocol Using the RSA Crypto Algorithm”, Cryptologia, Volume 7, number 2, pages 180-182, 1983