Solving Linear Equations Modulo Unknown Divisors: Revisited

Yao Lu[1,2], Rui Zhang[1] *, Liqiang Peng[1], and Dongdai Lin[1]

[1] State Key Laboratory of Information Security (SKLOIS),
Institute of Information Engineering, Chinese Academy of Sciences,
Beijing 100093, China
[2] The University of Tokyo, Japan
lywhhit@gmail.com, {r-zhang,pengliqiang,ddlin}@iie.ac.cn

Abstract. We revisit the problem of finding small solutions to a collection of linear equations modulo an unknown divisor p for a known composite integer N. In CaLC 2001, Howgrave-Graham introduced an efficient algorithm for solving univariate linear equations; since then, two forms of multivariate generalizations have been considered in the context of cryptanalysis: modular multivariate linear equations by Herrmann and May (Asiacrypt'08) and simultaneous modular univariate linear equations by Cohn and Heninger (ANTS'12). Their algorithms have many important applications in cryptanalysis, such as factoring with known bits problem, fault attacks on RSA signatures, analysis of approximate GCD problem, etc.

In this paper, by introducing multiple parameters, we propose several generalizations of the above equations. The motivation behind these extensions is that some attacks on RSA variants can be reduced to solving these generalized equations, and previous algorithms do not apply. We present new approaches to solve them, and compared with previous methods, our new algorithms are more flexible and especially suitable for some cases. Applying our algorithms, we obtain the best analytical/experimental results for some attacks on RSA and its variants, specifically,

- We improve May's results (PKC'04) on small secret exponent attack on RSA variant with moduli $N = p^r q$ ($r \geq 2$).
- We experimentally improve Boneh et al.'s algorithm (Crypto'98) on factoring $N = p^r q$ ($r \geq 2$) with known bits problem.
- We significantly improve Jochemsz-May' attack (Asiacrypt'06) on Common Prime RSA.
- We extend Nitaj's result (Africacrypt'12) on weak encryption exponents of RSA and CRT-RSA.

Keywords: Lattice-based analysis, Linear modular equations, RSA

* Corresponding author.

1 Introduction

Lattice-based cryptanalysis is a very useful tool in various cryptographic systems, e.g., historically, it was used to break the Merkle-Hellman knapsack cryptosystem [34]. The basic idea of the lattice-based approach is that if the system parameters of the target problem can be transformed into a basis of a certain lattice, one can find some short vectors in the desired lattice using dedicated algorithms, like the LLL-algorithm [20]. One may then hope that the secret key can be recovered once the solutions from these short vectors are extracted. Although in most cases this assumption is not rigorous in theory, it usually works well in practice.

In the above approach, a key step is to construct the desired lattice. In 1997, Coppersmith [5] presented a subtle lattice construction method, and used it to find small roots of modular equations of special forms. Since then, this approach has been widely applied in the analysis of RSA. Among them, one of the most important applications is to solve approximate integer common divisor problem (ACDP), namely, given two integers that are near-multiples of a hidden integer, output that hidden integer. We note that ACDP was first introduced by Howgrave-Graham [15], which in turn has many important applications such as building fully homomorphic cryptosystems [37].

Let us briefly explain Howgrave-Graham's method. First, one reduces ACDP to solving a univariate modular polynomial:

$$f(x) = x + a \bmod p$$

where a is a given integer, and p ($p \geq N^{\beta}$ for some $0 < \beta \leq 1$) is unknown that divides the known modulus N. Then he proposed a polynomial-time algorithm to find small roots of the univariate polynomial over integer. Note that this type of polynomial can also be applied in other RSA-related problems, such as factoring with known bits problem [21].

In 2003, May [21] generalized Howgrave-Graham's strategy by using a univariate linear polynomial to an arbitrary monic modular polynomial of degree δ, i.e. $f(x) = x^{\delta} + a_{\delta-1}x^{\delta-1} + \ldots + a_0 \bmod p$ where $\delta \geq 1$. As an important application, this algorithm can be used to solve the problem of factoring with known bits on Takagi's moduli $N = p^r q$ ($r > 1$) [2].

In Asiacrypt'08, Herrmann and May [12] extended the univariate linear modular polynomial to polynomials with an arbitrary number of n variables. They presented a polynomial-time algorithm to find small roots of linear modular-polynomials

$$f(x_1, \ldots, x_n) = a_0 + a_1 x_1 + \cdots + a_n x_n \bmod p$$

where p is unknown and divides the known modulus N. Naturally, they applied their results to the problem of factoring with known bits for RSA modulus $N = pq$ where those unknown bits might spread across arbitrary number of blocks of p. Besides, Herrmann-May's algorithm also can be used to cryptanalyze Multi-Prime Φ-Hiding Assumption [11,19], and attack CRT-RSA signatures [6,7].

On the other hand, in 2012, Cohn and Heninger [4] generalized Howgrave-Graham's equations to the simultaneous modular univariate linear equations

$$\begin{cases} f_1(x_1) = a_1 + x_1 = 0 \bmod p \\ f_2(x_2) = a_2 + x_2 = 0 \bmod p \\ \qquad\qquad \vdots \\ f_n(x_n) = a_n + x_n = 0 \bmod p \end{cases} \tag{1}$$

where $a_1, \ldots, a_n$ are given integers, and p ($p \geq N^\beta$ for some $0 < \beta < 1$) is an unknown factor of known modulus N. These equations have many applications in public-key cryptanalysis. For example, in 2010, van Dijk et al. [37] introduced fully homomorphic encryption over the integers, which the security of their scheme is based on the hardness of solving Equation (1). In 2011, Sarkar and Maitra [32] investigated implicit factorization problem [24] by solving Equations (1). In 2012, Fouque et al. [10] proposed fault attacks on CRT-RSA signatures, which can also be reduced to solving Equations (1).

1.1 Our Contributions

In this paper, we focus on the following three types of extensions of previous equations.

The first is an extension of Herrmann-May's equation, described in Section 3, we focus on the equations

$$f(x_1, x_2, \ldots, x_n) = a_0 + a_1 x_1 + \cdots + a_n x_n \bmod p^v \tag{2}$$

for some unknown divisor p^v ($v \geq 1$) and known composite integer N ($N \equiv 0 \bmod p^u$, $u \geq 1$). Here u, v are positive integers. Note that if $u = 1, v = 1$, that is exactly Herrmann-May's equation [12].

The second is a special case of Equations (2): $a_0 = 0$, described in Section 4.

The last is a generalized version of Equations (1), described in Section 5; we focus on the equations

$$\begin{cases} f_1(x_1) = a_1 + x_1 = 0 \bmod p^{r_1} \\ f_2(x_2) = a_2 + x_2 = 0 \bmod p^{r_2} \\ \qquad\qquad \vdots \\ f_n(x_n) = a_n + x_n = 0 \bmod p^{r_n} \end{cases} \tag{3}$$

where p ($p \geq N^\beta$ for some $0 < \beta < 1$) is unknown that satisfies $N = 0 \bmod p^r$ and $a_1, \ldots, a_n$, $r, r_1, \ldots, r_n$ are given integers. Here $r, r_1, \ldots, r_n$ are positive integers. Note that if $r = r_1 = \cdots = r_n = 1$, that is exactly Equations (1).

Notice that our generalized equations employ many parameters. The reason why we introduce these parameters is based on the fact that some attacks on RSA variants (such as Takagi's RSA variant [35]) can be reduced to solving this kind of equations. However, previous algorithms [23,12,4] do not seem to work in this

situation. The difficulty lies in how to wisely embed this algebraic information in the lattice construction.

We solve the above equations by introducing new techniques. More precisely, we present a novel way to select appropriate polynomials in constructing desired lattice. Compared with previous algorithms, our algorithms are more flexible and especially suitable for some cases. Applying our algorithms, we obtain the best analytical/experimental results for some attacks on RSA and its variants. We elaborate them below. We further conjecture that our new algorithms may find new applications in various other contexts.

Small Secret Exponent Attack on Multi-Power RSA. In multi-power RSA algorithm, suppose that the public key is (N, e), where $N = p^r q$ for some fixed $r \geq 2$ and p, q are of the same bit-size. The secret key d satisfies $ed \equiv 1 \bmod \phi(N)$, where $\phi(N)$ is Euler's ϕ-function. In Crypto'99, Takagi [35] showed that when the secret exponent $d < N^{\frac{1}{2(r+1)}}$, one can factorize N. Later in PKC'04, May [22] improved Takagi's bound to $N^{\max\{\frac{r}{(r+1)^2}, \frac{(r-1)^2}{(r+1)^2}\}}$. In this paper, we further improve May's bound to $N^{\frac{r(r-1)}{(r+1)^2}}$, which is better than May's result when $r > 2$, and is also independent of the value of public exponent e. Similar as [22], our result also directly implies an improved partial key exposure attack for secret exponent d with known most significant bits (MSBs) or least significant bits (LSBs). Our improvements are based on our algorithm of solving the first type equations, with the observation that $\gcd(ed - 1, N) = p^{r-1}$ but $N \equiv 0 \bmod p^r$.

Factoring Multi-Power Moduli with Known Bits. In 1999, Boneh et al. [2] extended factoring with high bits problem to moduli of the form $N = p^r q (r \geq 2)$. They showed that this moduli can be factored in polynomial-time in the bit-length of N if $r = \Omega(\sqrt{\frac{\log N}{\log \log N}})$. Applying our algorithm of solving the first type equations, we can directly get another method to settle the problem of [2]. Though we can not get an asymptotic improvement, in practice, expecially for large r, our new method performs better than [2].

Weak Encryption Exponents of RSA and CRT-RSA. In Africacrypt'12, Nitaj [26] presented some attacks on RSA and CRT-RSA (the public exponent e and the private CRT-exponents d_p and d_q satisfy $ed_p \equiv 1 \bmod (p-1)$ and $ed_q \equiv 1 \bmod (q-1)$). His attacks are based on Herrmann-May's technique [12] for finding small solutions of modular equations. In particular, he reduced his attacks to solving bivariate linear modular equations modulo unknown divisors: $ex + y \equiv 0 \bmod p$ for some unknown p that divides the known modulus N. Noticing that his equations are homogeneous, we can improve his results with our algorithm of solving second type equations.

Small Secret Exponent Attack on Common Prime RSA. We give a simple but effective attack on an RSA variant called Common Prime RSA. This

variant was originally introduced by Wiener [38] as a countermeasure for his continued fraction attack. He suggested to choose p and q such that $p-1$ and $q-1$ share a large common factor. In 2006, Hinek [13] revisited the security of Common Prime RSA, in the same year, Jochemsz and May [17] proposed a heuristic attack, and showed that parts of key space suggested by Hinek is insecure. In this paper, we further improve Jochemsz-May's bound by using our algorithm of solving third type equations.

Experimental Results. For all these attacks, we carry out experiments to verify the validity of our algorithms. These experimental results show that our attacks are effective.

2 Preliminary

In 1982, Lenstra, Lenstra and Lovász proposed the *LLL*-algorithm [20] that can find vectors in polynomial-time whose norm is small enough to satisfy the following condition.

Lemma 1 (LLL [20]). *Let $\mathcal{L}$ be a lattice of dimension w. Within polynomial-time, LLL-algorithm outputs a set of reduced basis vectors v_i, $1 \leqslant i \leqslant w$ that satisfies*

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

In practice, it is widely known that the *LLL*-algorithm tends to output the vectors whose norms are much smaller than theoretically predicted.

In 1997, Coppersmith [5] described a lattice-based technique to find small roots of modular and integer equations. Later, Howgrave-Graham [14] reformulated Coppersmith's ideas of finding modular roots. The main idea of Coppersmith's method is to reduce the problem of finding small roots of $f(x_1, \ldots, x_n) \bmod N$ to finding roots over the integers. Therefore, one can construct a collection of polynomials that share a common root modulo N^m for some well-chosen integer m. Then one can construct a lattice by defining a lattice basis via these polynomial's coefficient vectors. Using lattice basis reduction algorithms (like *LLL*-algorithm [20]), one can find a number of linear equations with sufficiently small norm. Howgrave-Graham [14] showed a sufficient condition to quantify the term sufficiently small. Next we review this useful lemma.

Let $g(x_1, \cdots, x_k) = \sum_{i_1, \cdots, i_k} a_{i_1, \cdots, i_k} x_1^{i_1} \cdots x_k^{i_k}$. We define the norm of g by the Euclidean norm of its coefficient vector: $||g||^2 = \sum_{i_1, \cdots, i_k} a_{i_1, \cdots, i_k}^2$.

Lemma 2 (Howgrave-Graham [14]). *Let $g(x_1, \cdots, x_k) \in \mathbb{Z}[x_1, \cdots, x_k]$ be an integer polynomial that consists of at most w monomials. Suppose that*

1. *$g(y_1, \cdots, y_k) = 0 \bmod p^m$ for $| y_1 | \leqslant X_1, \cdots, | y_k | \leqslant X_k$ and*
2. *$||g(x_1 X_1, \cdots, x_k X_k)|| < \frac{p^m}{\sqrt{w}}$*

Then $g(y_1, \cdots, y_k) = 0$ holds over integers.

Combining Lemma 1 and Lemma 2, we can get following theorem.

Theorem 1 (Coppersmith [5], May [23]). *Let N be an integer of unknown factorization, which has a divisor $p \geq N^{\beta}$, $0 < \beta \leq 1$. Let $f(x)$ be a univariate monic polynomial of degree δ. Then we can find in time $\mathcal{O}(\epsilon^{-7}\delta^5 \log^9 N)$ all solutions x_0 for the equation*

$$f(x) = 0 \bmod p \quad \textit{with} \quad |x_0| \leq N^{\frac{\beta^2}{\delta} - \epsilon}.$$

Additionally sometimes our attacks rely on a well-known assumption which was widely used in the literatures [1,9,12].

Assumption 1 *The lattice-based construction yields algebraically independent polynomials. The common roots of these polynomials can be efficiently computed using the* Gröbner basis technique.

Note that the time complexity of Gröbner basis computation is in general doubly exponential in the degree of the polynomials.

We would like to point out that our subsequent complexity considerations solely refer to our lattice basis reduction algorithm, that turns the polynomial $f(x_1, \ldots, x_n) \bmod N$ into the number of n polynomials over the integers. We assume that the running time of the Gröbner basis computation is negligible compared to the time complexity of the LLL-algorithm, since in general, our algorithm yields more than the number of n polynomials, so one can make use of these additional polynomials to speed up the Gröbner basis computation.

3 The First Type of Equations

In this section, we address how to solve $f_1(x) = a_0 + a_1 x \bmod p^v$ $(v \geq 1)$ for some unknown p where p^u divides a known modulus N (i.e. $N \equiv 0 \bmod p^u$, $u \geq 1$). In particular, Howgrave-Graham's result [15] can be viewed as a special case of our algorithm when $u = 1$, $v = 1$.

3.1 Our Main Result

Theorem 2. *For every $\epsilon > 0$, let N be a sufficiently large composite integer (of unknown factorization) with a divisor p^u $(p \geq N^{\beta}, u \geq 1)$. Let $f_1(x) \in \mathbb{Z}[x]$ be a univariate linear polynomial whose leading coefficient is coprime to N. Then one can find all the solutions y of the equation $f_1(x) = 0 \bmod p^v$ with $v \geq 1$, $|y| \leq N^{\gamma}$ if $\gamma < uv\beta^2 - \epsilon$. The time complexity is $\mathcal{O}(\epsilon^{-7} v^2 \log^2 N)$.*

Proof. Consider the following univariate linear polynomial:

$$f_1(x) = a_0 + a_1 x \bmod p^v$$

where N is known to be a multiple of p^u for known u and unknown p. Here we assume that $a_1 = 1$, since otherwise we can multiply f_1 by $a_1^{-1} \bmod N$. Let $f(x) = a_1^{-1} f_1(x) \bmod N$.

$$
\begin{array}{c}
N^4 \\ fN^4 \\ f^2N^3 \\ f^3N^2 \\ f^4N^2 \\ f^5N \\ f^6 \\ f^7 \\ f^8
\end{array}
\begin{pmatrix}
N^4 & & & & & & & & \\
* & XN^4 & & & & & & & \\
* & * & X^2N^3 & & & & & & \\
* & * & * & X^3N^2 & & & & & \\
* & * & * & * & X^4N^2 & & & & \\
* & * & * & * & * & X^5N & & & \\
* & * & * & * & * & * & X^6 & & \\
* & * & * & * & * & * & * & X^7 & \\
* & * & * & * & * & * & * & * & X^8
\end{pmatrix}
$$

Fig. 1. The matrix for the case $\beta = 0.25$, $u = 3$, $v = 2$, $t = 6$, $m = 8$

We define a collection of polynomials as follows:

$$g_k(x) := f^k(x) N^{\max\{\lceil \frac{v(t-k)}{u} \rceil, 0\}}$$

for $k = 0, \ldots, m$ and integer parameters t and m with $t = \tau m$ $(0 \leq \tau < 1)$, which will be optimized later. Note that for all k, $g_k(y) \equiv 0 \bmod p^{vt}$.

Let $X := N^{uv\beta^2 - \epsilon} (= N^{\gamma})$ be the upper bound on the desired root y. We will show that this bound can be achieved for any chosen value of ϵ by ensuring that $m \geq m^* := \lceil \frac{\beta(2u+v-uv\beta)}{\epsilon} \rceil - 1$

We build a lattice $\mathcal{L}$ of dimension $d = m + 1$ using the coefficient vectors of $g_k(xX)$ as basis vectors. We sort these polynomials according to the ascending order of g, i.e., $g_k < g_l$ if $k < l$. Fig. 1 shows an example for the parameters $\beta = 0.25, u = 3, v = 2, t = 6, m = 8$.

From the triangular matrix of the lattice basis, we can compute the determinant as the product of the entries on the diagonal as $\det(\mathcal{L}) = X^s N^{s_N}$ where

$$
\begin{aligned}
s &= \sum_{k=0}^{m} k = \frac{m(m+1)}{2} \\
s_N &= \sum_{k=0}^{t-1} \lceil \frac{v(t-k)}{u} \rceil = \sum_{k=0}^{t-1} \left(\frac{v(t-k)}{u} + c_k \right) = \frac{v\tau m(\tau m + 1)}{2u} + \sum_{k=0}^{t-1} c_k
\end{aligned}
$$

Here we rewrite $\lceil \frac{v(t-k)}{u} \rceil$ as $\left(\frac{v(t-k)}{u} + c_k \right)$ where $c_k \in [0, 1)$. To obtain a polynomial with short coefficients that contains all small roots over integer, we apply LLL-basis reduction algorithm to the lattice $\mathcal{L}$. Lemma 1 gives us an upper bound on the norm of the shortest vector in the LLL-reduced basis; if the bound is smaller than the bound given in Lemma 2, we can obtain the desired polynomial. We require the following condition:

$$2^{\frac{d-1}{4}} \det(\mathcal{L})^{\frac{1}{d}} < \frac{N^{v\beta\tau m}}{\sqrt{d}}$$

where $d = m + 1$. We plug in the values for $\det(\mathcal{L})$ and d, and obtain

$$2^{\frac{m(m+1)}{4}}(m+1)^{\frac{m+1}{2}}X^{\frac{m(m+1)}{2}} < N^{v\beta\tau m(m+1)-\frac{v\tau m(\tau m+1)}{2u}-\sum_{k=0}^{t-1}c_k}$$

To obtain the asymptotic bound, we let m grow to infinity. Note that for sufficiently large N the powers of 2 and $m + 1$ are negligible. Thus, we only consider the exponent of N. Then we have

$$X < N^{2v\beta\tau-\frac{v\tau(\tau m+1)}{u(m+1)}-\frac{2\sum_{k=0}^{t-1}c_k}{m(m+1)}}$$

Setting $\tau = u\beta$, and noting that $\sum_{k=0}^{t-1} c_k \leq t$ [3], the exponent of N can be lower bounded by

$$uv\beta^2 - \frac{v\beta(1-u\beta)}{m+1} - \frac{2u\beta}{m+1}$$

We appropriate the negative term $\frac{*}{m+1}$ by $\frac{*}{m}$ and obtain

$$uv\beta^2 - \frac{\beta(2u+v-uv\beta)}{m}$$

Enduring that $m \geq m^*$ will then gurantee that X satifies the required bound for the chosen value of ϵ.

The running time of our method is dominated by LLL-algorithm, which is polynomial in the dimension of the lattice and in the maximal bit-size of the entries. We have a bound for the lattice d

$$d = m + 1 \geq \lceil \frac{\beta(2u+v-uv\beta)}{\epsilon} \rceil$$

Since $u\beta < 1$, then we obtain $d = \mathcal{O}(\epsilon^{-1})$. The maximal bit-size of the entries is bounded by

$$\max\{\frac{vt}{u}\log(N), duv\beta^2\log(N)\} = \max\{v\beta d\log(N), duv\beta^2\log(N)\}$$

Since $u\beta < 1$ and $d = \mathcal{O}(\epsilon^{-1})$, the bit-size of the entries can be upperbounded by

$$\max\{\mathcal{O}(v\beta\epsilon^{-1})\log(N), \mathcal{O}(v\beta\epsilon^{-1})\log(N)\} = \mathcal{O}(v\epsilon^{-1}\log(N))$$

Nguyen and Stehlé [25] proposed a modified version of the LLL-algorithm called L^2-algorithm. The L^2-algorithm achieves the same approximation quality for a shortest vectors as the LLL-algorithm, but has an improved worst case running time anlaysis. Its running time is $\mathcal{O}(d^5(d + \log b_d)\log b_d)$, where $\log b_d$ is the maximal bit-size of an entry in lattice. Thus, we can obtain the running time of our algorithm

$$\mathcal{O}\left(\left(\frac{1}{\epsilon}\right)^5\left(\frac{1}{\epsilon}+\frac{v\log N}{\epsilon}\right)\frac{v\log N}{\epsilon}\right)$$

Therefore, the running time of our algorithm is $\mathcal{O}(\epsilon^{-7}v^2\log^2 N)$. Eventually, the vector output by LLL-algorithm gives a univariate polynomial $g(x)$ such that $g(y) = 0$, and one can find the root of $g(x)$ over the integers. □

[3] This estimation is rough, we can do it more precisely for specific parameters u, v. For example, for $v = 1$, we can get $\sum_{k=0}^{t-1} c_k \leq \frac{t}{2} + 1$

Extension to Arbitrary Degree. We can generalize the result of Theorem 2 to univariate polynomials with arbitrary degree.

Theorem 3. *For every $\epsilon > 0$, let N be a sufficiently large composite integer (of unknown factorization) with a divisor p^u ($p \geq N^\beta$, $u \geq 1$). Let $f_1(x) \in \mathbb{Z}[x]$ be a univariate polynomial of degree δ whose leading coefficient is coprime to N. Then one can find all the solutions y of the equation $f_1(x) = 0 \pmod{p^v}$ with $v \geq 1$, $|y| \leq N^\gamma$ if $\gamma < \frac{uv\beta^2}{\delta} - \epsilon$. The time complexity is $\mathcal{O}(\epsilon^{-7}\delta^5 v^2 \log^2 N)$.*

In the proof of Theorem 3, we use the following collection of polynomials:

$$g_k(x) := x^j f^k(x) N^{\max\{\lceil \frac{v(t-k)}{u} \rceil, 0\}}$$

for $k = 0, \ldots, m$, $j = 0, \ldots, \delta - 1$ and integer parameters t and m with $t = \tau m$ ($0 \leq \tau < 1$). The rest of the proof is the same as Theorem 2. We omit it here.

Specifically, the result in [23] can be viewed as a special case of our algorithm when $u = v$.

Extension to More Variables. We also generalize the result of Theorem 2 from univariate linear equations to an arbitrary number of n variables $x_1, \ldots, x_n$ ($n \geq 2$).

Proposition 1. *For every $\epsilon > 0$, let N be a sufficiently large composite integer (of unknown factorization) with a divisor p^u ($p \geq N^\beta$, $u \geq 1$). Furthermore, let $f_1(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ be a monic linear polynomial in $n (n \geq 2)$ variables. Under Assumption 1, we can find all the solutions $(y_1, \ldots, y_n)$ of the equation $f_1(x_1, \ldots, x_n) = 0 \pmod{p^v}$ with $v \geq 1$, $|y_1| \leq N^{\gamma_1}, \ldots |y_n| \leq N^{\gamma_n}$ if*

$$\sum_{i=1}^{n} \gamma_i < \frac{v}{u}\left(1 - (1-u\beta)^{\frac{n+1}{n}} - (n+1)(1-u\beta)\left(1 - \sqrt[n]{1-u\beta}\right)\right) - \epsilon$$

The running time of the algorithm is polynomial in ϵ^{-n} and $\epsilon^{-n} \log N$.

Proof. We define the following collection of polynomials which share a common root modulo p^{vt}:

$$g_{i_2,\ldots,i_n,k} = x_2^{i_2} \cdots x_n^{i_n} f_1^k N^{\max\{\lceil \frac{v(t-k)}{u} \rceil, 0\}}$$

for $k = 0, ..., m$ where $i_j \in \{0, \ldots, m\}$ such that $\sum_{j=2}^{n} i_j \leq m - k$, and the integer parameter $t = \tau m$ has to be optimized. The idea behind the above transformation is that we try to eliminate powers of N in the diagonal entries in order to keep the lattice determinant as small as possible.

Next we can construct the lattice $\mathcal{L}$ using the similar method of Herrmann-May [12], therefore, the lattice has triangular form, then the determinant $\det(\mathcal{L})$ is then simply the product of the entries on the diagonal:

$$\det(\mathcal{L}) = \prod_{i=1}^{n} X_i^{s_{x_i}} N^{s_N}$$

Let d denote the dimension of $\mathcal{L}$, $t = r \cdot h + c$ ($h, c \in \mathbb{Z}$ and $0 \leq c < r$). A straightforward but tedious computation yields that

$$\begin{aligned} s_{x_i} &= \binom{m+n}{m-1} = \frac{1}{(n+1)!} m^{n+1} + o(m^{n+1}) \\ s_N &= \sum_{k=0}^{t-1} \sum_{0 \leq \sum_{j=2}^{n} i_j \leq m-k} \lceil \frac{v(t-k)}{u} \rceil \\ &= \frac{v}{u} \frac{(n+1)\tau - 1 + (1-\tau)^{n+1}}{(n+1)!} m^{n+1} + o(m^{n+1}) \\ d &= \binom{m+n}{m} = \frac{1}{n!} m^n + o(m^n) \end{aligned}$$

To obtain the number of n polynomial with short coefficients that contains all small roots over integer, we apply LLL-basis reduction algorithm to the lattice $\mathcal{L}$. Combining Lemma 1 with Lemma 2, we require the following condition:

$$2^{\frac{d(d-1)}{4(d+1-n)}} \det(\mathcal{L})^{\frac{1}{d-n+1}} < \frac{N^{v\beta\tau m}}{\sqrt{d}}$$

Let $X_i = N^{\gamma_i} (1 \leq i \leq n)$. Combining the values with the above condition, we obtain

$$\sum_{i=1}^{n} \gamma_i < \frac{v}{u} \left(1 - (1-\tau)^{n+1}\right) - \tau v (n+1)(\frac{1}{u} - \beta) - \epsilon$$

By setting $\tau = 1 - \sqrt[n]{1 - u\beta}$, the condition reduces to

$$\sum_{i=1}^{n} \gamma_i < \frac{v}{u} \left(1 - (1-u\beta)^{\frac{n+1}{n}} - (n+1)(1-u\beta)\left(1 - \sqrt[n]{1-u\beta}\right)\right) - \epsilon$$

The running time is dominated by the time to run LLL-lattice reduction on a basis matrix of dimension d and bit-size of the entries. Since $d = \mathcal{O}(\frac{m^n}{n!})$ and the parameter m depends on ϵ^{-1} only, therefore, our approach is polynomial in $\log N$ and ϵ^{-n}. Besides, our attack relies on Assumption 1.

□

3.2 Analysis of Multi-Power RSA

We apply our algorithm to analyze an RSA variant, namely multi-power RSA, with moduli $N = p^r q$ ($r \geq 2$). Compared to the standard RSA, the multi-power RSA is more efficient in both key generation and decryption. Besides, moduli of this type have been applied in many cryptographic designs, e.g., the Okamoto-Uchiyama cryptosystem [27], or better known via EPOC and ESIGN [8], which uses the modulus $N = p^2 q$.

Using our algorithm of Theorem 2, we give two attacks on multi-power RSA: small secret exponent attack and factoring with known bits.

Table 1. Comparisons of May's bound, Sarkar's bound and ours on δ

r	2	3	4	5	6	7	8	9
May's bound	0.22	0.25	0.36	0.44	0.51	0.56	0.60	0.64
Sarkar's bound [31,30]	**0.39**	**0.46**	**0.50**	0.54	0.57	0.51	0.53	0.54
Our bound	0.22	0.37	0.48	**0.55**	**0.61**	**0.65**	**0.69**	**0.72**

Small Secret Exponent Attack on Multi-Power RSA. There are two variants of multi-power RSA. In the first variant $ed \equiv 1 \bmod p^{r-1}(p-1)(q-1)$, while in the second variant $ed \equiv 1 \bmod (p-1)(q-1)$. In [16], the authors proved that the second variant is vulnerable when $d < N^{\frac{2-\sqrt{2}}{r+1}}$.

In this section, we focus on the first variant. In Crypto'99, Takagi [35] proved that when the decryption exponent $d < N^{\frac{1}{2(r+1)}}$, one can factorize N in polynomial-time. Later, in PKC'04, May [22] improved Takagi's bound to $N^{\max\{\frac{r}{(r+1)^2}, \frac{(r-1)^2}{(r+1)^2}\}}$. Based on the technique of Theorem 2, we can further improve May's bound to $N^{\frac{r(r-1)}{(r+1)^2}}$.

Theorem 4. *Let $N = p^r q$, where $r \geq 2$ is a known integer and p, q are primes of the same bit-size. Let e be the public key exponent and d be the private key exponent, satisfying $ed \equiv 1 \bmod \phi(N)$. For every $\epsilon > 0$, suppose that*

$$d < N^{\frac{r(r-1)}{(r+1)^2} - \epsilon}$$

then N can be factored in polynomial-time.

Proof. Since $\phi(N) = p^{r-1}(p-1)(q-1)$, we have the equation $ed - 1 = kp^{r-1}(p-1)(q-1)$ for some $k \in \mathbb{N}$. Then we want to find the root $y = d$ of the polynomial

$$f_1(x) = ex - 1 \bmod p^{r-1}$$

with the known multiple (of unknown divisor p) N ($N \equiv 0 \bmod p^r$). Let $d \approx N^{\delta}$. Applying Theorem 2, setting $\beta = \frac{1}{r+1}$, $u = r$, $v = r-1$, we obtain the final result $\delta < \frac{r(r-1)}{(r+1)^2} - \epsilon$ □

Recently, Sarkar [31,30] improved May's bound for modulus $N = p^r q$, however, unlike our method, his method can not applied for public key exponents e of arbitrary size. In addition, we get better experimental results for the case of $r > 2$ (see Section 3.2).

For small r, we provide the comparison of May's bound, Sarkar's bound, and our bound on δ in Table 1. Note that for $r = 2$, we obtain the same result as May's bound.

Partial Key-Exposure Attacks on Multi-Power RSA. Similar to the results of [22], the new attack of Theorem 4 immediately implies partial key exposure attacks for d with known MSBs/LSBs. Following we extend the approach of Theorem 4 to partial key exposure attacks.

Table 2. Experimental results of the attack from Theorem 4

N (bits)	r	e(bits)	d-pred(bits)	(m,t)	dim($\mathcal{L}$)	d-exp(bits)	δ	time(sec)
1536	2	1536	341	$(30,20)$	31	318	0.207	3155.687
2048	3	2048	768	$(20,15)$	21	706	0.345	749.167
2048	3	4096	768	$(20,15)$	21	706	0.345	745.170
2048	3	2048	768	$(40,30)$	41	735	0.359	37800.462
2560	4	2560	1228	$(20,16)$	21	1136	0.444	1245.754
2560	4	2560	1228	$(30,24)$	31	1167	0.456	12266.749

Theorem 5 (MSBs). *Let $N = p^r q$, where $r \geq 2$ is a known integer and p, q are primes of the same bit-size. Let e be the public key exponent and d be the private key exponent, satisfying $ed = 1 \bmod \phi(N)$. For every $\epsilon > 0$, given $\tilde{d}$ such that $|d - \tilde{d}| < N^{\frac{r(r-1)}{(r+1)^2} - \epsilon}$, then N can be factored in polynomial-time.*

Proof. We have that

$$e(d - \tilde{d}) + e\tilde{d} - 1 \equiv 0 \bmod p^{r-1}$$

Then we want to find the root $y = d - \tilde{d}$ of the polynomial

$$f_1(x) = ex + e\tilde{d} - 1 \bmod p^{r-1}$$

with the known multiple (of unknown divisor p) N ($N \equiv 0 \bmod p^r$). Applying Theorem 2, setting $\beta = \frac{1}{r+1}$, $u = r$, $v = r - 1$, we obtain the final result. □

Theorem 6 (LSBs). *Let $N = p^r q$, where $r \geq 2$ is a known integer and p, q are primes of the same bit-size. Let e be the public key exponent and d be the private key exponent, satisfying $ed = 1 \bmod \phi(N)$. For every $\epsilon > 0$, given d_0, M with $d = d_0 \bmod M$ and $M > N^{\frac{3r+1}{(r+1)^2} + \epsilon}$, then N can be factored in polynomial-time.*

Proof. Rewrite $d = d_1 M + d_0$, then we have

$$ed_1 M + ed_0 - 1 \equiv 0 \bmod p^{r-1}$$

Then we want to find the root $y = d_1$ of the polynomial

$$f_1(x) = eMx + ed_0 - 1 \bmod p^{r-1}$$

with the known multiple (of unknown divisor p) N ($N \equiv 0 \bmod p^r$). Applying Theorem 2 and setting $\beta = \frac{1}{r+1}$, $u = r$, $v = r - 1$, we obtain the final result. □

We have implemented our algorithm in Magma 2.11 computer algebra system on our PC with Intel(R) Core(TM) Duo CPU (2.53GHz, 1.9GB RAM Windows 7). Table 2 shows the experimental results for multi-power RSA modulus N with 512-bit primes p, q. We compute the number of bits that one should theoretically be able to attack for d (column d-pred in Table 2). In all the listed experiments,

we can recover the factorization of N. Note that our attack is also effective for large e.

In [31], for 1024-bit $N = p^3q$, Sarkar considered $\delta = 0.27$ using a lattice with dimension 220, while we can achieve $\delta = 0.359$ using a lattice with dimension 41. Besides, Sarkar also stated that "for $r = 4, 5$, lattice dimension in our approach becomes very large to achieve better results. Hence in these cases we can not present experiment results to show the improvements over existing results." In Table 2, we can see that our experimental results are better than Sarkar's for $r > 2$.

Factoring Multi-Power Moduli with Known Bits. In 1985, Rivest and Shamir [28] first introduced the factoring with high bits known problem, they presented an algorithm that factors $N = pq$ given $\frac{2}{3}$-fraction of the bits of p. Later, Coppersmith [5] gave a improved algorithm when half of the bits of p are known. In 1999, Boneh, Durfee and Howgrave-Graham [2] (referred as BDH method) extended Coppersmith's results to moduli $N = p^r q (r \geq 2)$. Basically, they considered the scenario that a few MSBs of the prime p are known to the attacker. Consider the univariate polynomial

$$f(x) = (\tilde{p} + x)^r \bmod p^r$$

For simplicity, we assume that p and q are of the same bit-size. Using the algorithm of Theorem 1, Boneh et al. showed that they can recover all roots x_0 with

$$|x_0| \leq N^{\frac{\beta^2}{\delta} - \epsilon} = N^{\frac{r}{(r+1)^2} - \epsilon}$$

in time $\mathcal{O}(\epsilon^{-7} \log^2 N)$ [4]. Thus we need a $\frac{1}{r+1}$-fraction of p in order to factor N in polynomial-time.

Applying our algorithm of Theorem 2, and setting $\beta = \frac{1}{r+1}, u = r, v = 1$, we can also find all roots x_0 with

$$|x_0| \leq N^{uv\beta^2 - \epsilon} = N^{\frac{r}{(r+1)^2} - \epsilon}$$

in time $\mathcal{O}(\epsilon^{-7} \log^2 N)$.

Note that we obtain the same asymptotic bound and running time complexity as BDH method. But, as opposed to BDH method, our algorithm is more flexible in choosing the lattice dimension. For example, in the case of $r = 10$, BDH method only works on the lattice dimension of $11 * m$ ($m \in \mathbb{Z}^+$) while our method can work on any lattice dimension m ($m \in \mathbb{Z}^+$). Figure 2 shows a comparsion of these two methods in terms of the size of $\tilde{p}$ ($\tilde{p} = N^\gamma$) that can be achieved. We can see that to achieve the same γ, we require smaller lattice dimensions than BDH method. Our algorithm is especially useful for large r. Actually our lattice is the same to the lattice of BDH method if the lattice dimensions are $11 * m$ ($m \in \mathbb{Z}^+$).

[4] Since this univariate equation is very special: $f(x) = (x + a)^r$, in fact we can remove the quantity r^5 from the time complexity of Theorem 1.

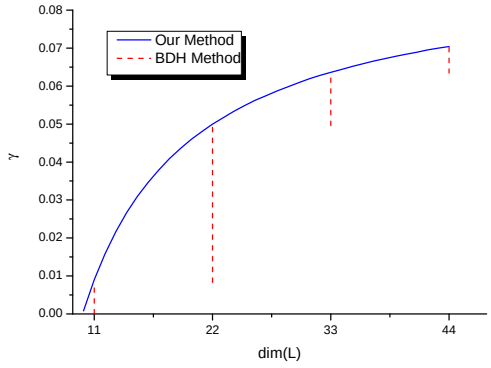

Fig. 2. Comparison of the achievable bound depending on the lattice dimension: the case of $r = 10$.

Table 3. Comparison of our experimental results with BDH method.

r	Theo.	Expt.	BDH method		Our method	
			Dim	Time (in seconds)	Dim	Time (in seconds)
5	84	164	30	112.914	26	29.281
5	84	134	48	2874.849	46	1343.683
10	46	186	44	670.695	34	259.298
10	46	166	44	1214.281	41	917.801

We also give some experimental results. Table 3 shows the experimental results for multi-power RSA modulus $N(N = p^r q)$ with 500-bit primes p, q. These experimental data confirmed our theoretical analysis. It is obvious that our method performs better than BDH method in practice.

4 The Second Type of Equations

In this section, we study the problem of finding small roots of homogeneous linear polynomials $f_2(x_1, x_2) = a_1 x_1 + a_2 x_2 \bmod p^v$ $(v \geq 1)$ for some unknown p where p^u divides a known modulus N (i.e. $N \equiv 0 \bmod p^u$, $u \geq 1$). Let (y_1, y_2) be a small solution of $f_2(x_1, x_2)$. We assume that we also know an upper bound $(X_1, X_2) \in \mathbb{Z}^2$ for the root such that $|y_1| \leq X_1, |y_2| \leq X_2$.

4.1 Our Main Result

Theorem 7. *For every $\epsilon > 0$, let N be a sufficiently large composite integer (of unknown factorization) with a divisor p^u $(p \geq N^\beta$, $u \geq 1)$. Let $f_2(x_1, x_2) \in \mathbb{Z}[x_1, x_2]$ be a homogeneous linear polynomial in two variables whose coefficients*

are coprime to N. Then one can find all the solutions (y_1, y_2) of the equation $f_2(x_1, x_2) = 0 \pmod{p^v}$ $(v \geq 1)$ with $\gcd(y_1, y_2) = 1$, $|y_1| \leq N^{\gamma_1}, |y_2| \leq N^{\gamma_2}$ if $\gamma_1 + \gamma_2 < uv\beta^2 - \epsilon$, and the time complexity of our algorithm is $\mathcal{O}(\epsilon^{-7} v^2 \log^2 N)$.

Proof. Since the proof is similar to that of Theorem 2, we only give the sketch here. Consider the linear polynomial:

$$f_2(x_1, x_2) = a_1 x_1 + a_2 x_2 \bmod p^v$$

where N is known to be a multiple of p^u for known u and unknown p. Here we assume that $a_1 = 1$, since otherwise we can multiply f_2 by $a_1^{-1} \bmod N$. Let

$$f(x_1, x_2) = a_1^{-1} f_2(x_1, x_2) \bmod N$$

Fix $m := \lceil \frac{\beta(2u+v-uv\beta)}{\epsilon} \rceil$, and define a collection of polynomials as follows:

$$g_k(x_1, x_2) := x_2^{m-k} f^k(x_1, x_2) N^{\max\{\lceil \frac{v(t-k)}{u} \rceil, 0\}}$$

for $k = 0, \ldots, m$ and integer parameters t and m with $t = \tau m$ $(0 \leq \tau < 1)$, which will be optimized later. Note that for all k, $g_k(y_1, y_2) \equiv 0 \bmod p^{vt}$.

Let $X_1, X_2 (X_1 = N^{\gamma_1}, X_2 = N^{\gamma_2})$ be upper bounds on the desired root (y_1, y_2), and define $X_1 X_2 := N^{uv\beta^2 - \epsilon}$. We build a lattice $\mathcal{L}$ of dimension $d = m+1$ using the coefficient vectors of $g_k(x_1 X_1, x_2 X_2)$ as basis vectors. We sort the polynomials according to the order as following: If $k < l$, then $g_k < g_l$.

From the triangular matrix of the lattice, we can easily compute the determinant as the product of the entries on the diagonal as $\det(\mathcal{L}) = X_1^{s_1} X_2^{s_2} N^{s_N}$ where

$$s_1 = s_2 = \sum_{k=0}^{m} k = \frac{m(m+1)}{2}$$

$$s_N = \sum_{k=0}^{t-1} \lceil \frac{v(t-k)}{u} \rceil = \sum_{k=0}^{t-1} \left(\frac{v(t-k)}{u} + c_k \right) = \frac{vt(t+1)}{2u} + \sum_{k=0}^{t-1} c_k$$

Here we rewrite $\lceil \frac{v(t-k)}{u} \rceil$ as $\left(\frac{v(t-k)}{u} + c_k \right)$ where $c_k \in [0, 1)$. Combining Lemma 2 and Lemma 1, after some calculations, we can get the final result

$$\gamma_1 + \gamma_2 \leq uv\beta^2 - \frac{\beta(2u + v - uv\beta)}{m}$$

Similar to Theorem 2, the time complexity of our algorithm is $\mathcal{O}(\epsilon^{-7} v^2 \log^2 N)$.

The vector output by *LLL*-algorithm gives a polynomial $f'(x_1, x_2)$ such that $f'(y_1, y_2) = 0$. Let $z = x_1/x_2$, any rational root of the form y_1/y_2 can be found by extracting the rational roots of $f'(z) = 1/x_2^m f'(x_1, y_1)$ with classical methods. □

Comparisons with Previous Methods. For $u = 1, v = 1$, the upper bound $\delta_1 + \delta_2$ of Theorem 7 is β^2, that is exactly May's results [21] on univariate linear polynomial $f(x) = x + a$. Actually the problem of finding a small root of homogeneous polynomial $f(x_1, x_2)$ can be transformed to find small rational roots of univariate linear polynomial $F(z)$ i.e. $F(\frac{x_2}{x_1}) = f(x_1, x_2)/x_1$ (the discussions of the small rational roots can be found on pp. 413 of Joux's book [18]).

Our result improves Herrmann-May's bound $3\beta - 2 + 2(1-\beta)^{\frac{3}{2}}$ up to β^2 if $a_0 = 0$. As a concrete example, for the case $\beta = 0.5$, our method improves the upper size of X_1X_2 from $N^{0.207}$ to $N^{0.25}$.

Another important work to mention is that in [3], Castagnos, Joux, Laguillaumie and Nguyen also considered homogeneous polynomials. Their algorithm can be directly applied to our attack scenario. They consider the following bivariate homogeneous polynomial

$$f(x_1, x_2) = (a_1x_1 + a_2x_2)^{\frac{u}{v}} \bmod p$$

However, their algorithm can only deal with the cases $\frac{u}{v} \in \mathbb{Z}$, and our algorithm is more flexible: specially, for $\frac{u}{v}$-degree polynomial with $2^{\frac{u}{v}}$ monomials (the dimension of lattice is $\frac{u}{v}m$), whereas our algorithm is for linear polynomial with two monomials (the dimension of lattice is m). Besides, in [3], they formed a lattice using the coefficients of $g(x, y)$ instead of $g(xX, yY)$. This modification enjoys the benefits in terms of real efficiency, since their lattice has smaller determinant than in the classical bivariate approach. However, their algorithm fails when the solutions are significantly unbalanced ($X_1 \gg X_2$). We highlight the idea that the factor X, Y should not only be used to balance the size of different power of x, y but also to balance the variables x, y. That is why our algorithm is suitable for this unbalanced attack scenario.

Extension to More Variables. We generalize the result of Theorem 7 to an arbitrary number of n variables $x_1, \ldots, x_n$. The proof of the following result is similar to that for Proposition 1, so we state only the result itself.

Proposition 2. *For every $\epsilon > 0$, let N be a sufficiently large composite integer (of unknown factorization) with a divisor p^u ($p \geq N^\beta$, $u \geq 1$). Furthermore, let $f_2(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ be a homogeneous linear polynomial in $n (n \geq 3)$ variables. Under Assumption 1, we can find all the solutions $(y_1, \ldots, y_n)$ of the equation $f_2(x_1, \ldots, x_n) = 0 \mod p^v$ ($v \geq 1$) with $\gcd(y_1, \ldots, y_n) = 1$, $|y_1| \leq N^{\gamma_1}, \ldots |y_n| \leq N^{\gamma_n}$ if*

$$\sum_{i=1}^{n} \gamma_i < \frac{v}{u}\left(1 - (1-u\beta)^{\frac{n}{n-1}} - n(1-u\beta)\left(1 - \sqrt[n-1]{1-u\beta}\right)\right) - \epsilon$$

The running time of the algorithm is polynomial in $\log N$ and $\epsilon^{-n} \log N$.

4.2 Applications

In Africacrypt'12, Nitaj [26] presented a new attack on RSA. His attack is based on Herrmann-May's method [12] for finding small roots of a bivariate

linear equation. In particular, he showed that the public modulus N can be factored in polynomial-time for the RSA cryptosystem where the public exponent e satisfies an equation $ex + y \equiv 0 \pmod{p}$ with parameters x and y satisfying $ex + y \not\equiv 0 \pmod{N}$ $|x| < N^{\gamma}$ and $|y| < N^{\delta}$ with $\delta + \gamma \leq \frac{\sqrt{2}-1}{2}$.

Note that the equation of [26] is homogeneous, thus we can improve the upper bound of $\gamma + \delta$ using our result in Theorem 7. In [29], Sarkar proposed another method to extend Nitaj's weak encryption exponents. Here, the trick is to consider the fact that Nitaj's bound can be improved when the unknown variables in the modular equation are unbalanced (x and y are of different bit-size). In general, Sarkar's method is essentially Herrmann-May's method, whereas our algorithm is simpler (see Theorem 7). We present our result below.

Theorem 8. *Let $N = pq$ be an RSA modulus with $q < p < 2q$. Let e be a public exponent satisfying an equation $ex + y \equiv 0 \bmod p$ with $|x| < N^{\gamma}$ and $|y| < N^{\delta}$. If $ex + y \not\equiv 0 \bmod N$ and $\gamma + \delta \leq 0.25 - \epsilon$, N can be factored in polynomial-time.*

In [26], Nitaj also proposed a new attack on CRT-RSA. Let $N = pq$ be an RSA modulus with $q < p < 2q$. Nitaj showed that if $e < N^{\frac{\sqrt{2}}{2}}$ and $ed_p = 1 + k_p(p-1)$ for some d_p with $d_p < \frac{N^{\frac{\sqrt{2}}{4}}}{\sqrt{e}}$, N can be factored in polynomial-time. His method is also based on Herrmann-May's method. Similarly we can improve Nitaj's result in some cases using our idea as Theorem 7.

Theorem 9. *Let $N = pq$ be an RSA modulus with $q < p < 2q$. Let e be a public exponent satisfying $e < N^{0.75}$ and $ed_p = 1 + k_p(p-1)$ for some d_p with*

$$d_p < \frac{N^{\frac{0.75-\epsilon}{2}}}{\sqrt{e}}$$

Then, N can be factored in polynomial-time.

Proof. We rewrite the equation $ed_p = 1 + k_p(p-1)$ as

$$ed_p + k_p - 1 = k_p p$$

Then we focus on the equation modulo p

$$ex + y = 0 \bmod p$$

with a root $(x_0, y_0) = (d_p, k_p - 1)$. Suppose that $e = N^{\alpha}$, $d_p = N^{\delta}$, then we get

$$k_p = \frac{ed_p - 1}{p-1} < \frac{ed_p}{p-1} < N^{\alpha+\delta-0.5}$$

Applying Theorem 7 with the desired equation where $x_0 = d_p < N^{\delta}$ and $y_0 = k_p - 1 < N^{\alpha+\delta-0.5}$, setting $\beta = 0.5$, $u = 1$ and $v = 1$ we obtain

$$2\delta + \alpha < 0.75 - \epsilon$$

Note that $\gcd(x_0, y_0) = \gcd(d_p, k_p - 1) = 1$, $k_p < N^{\alpha+\delta-0.5} < N^{\alpha+2\delta-0.5} < N^{0.25} < p$, hence $ed_p + k_p - 1 \neq 0 \bmod N$. Then we can factorize N with $\gcd(N, ed_p + k_p - 1) = p$. □

Table 4. Experimental results for weak encryption exponents

N (bit)	r	d_p-pred(bits)	(m,t)	dim($\mathcal{L}$)	d_p-exp(bits)	Time(sec)
1024	1	128	$(6,3)$	7	110	0.125
1024	1	128	$(10,5)$	11	115	1.576
1024	1	128	$(30,15)$	31	124	563.632

Note that Theorem 9 requires the condition $e < N^{0.75}$ for $N = pq$, hence we cannot be using small CRT exponents both modulo p and modulo q. Our attack is valid for the case that the cryptographic algorithm has a small CRT-exponent modulo p, but a random CRT-exponent modulo q.

Experimental Results. Table 4 shows the experimental results for RSA modulus N with 512-bit primes p, q. In all of our experiments, we fix e's length as 512-bit, and so the scheme does not have a small CRT exponent modulo q. We also compute the number bits that one should theoretically be able to attack for d_p (column d_p-pred of Table 4).

That is actually the attack described in Theorem 9. In [26], the author showed that for a 1024-bit modulus N, the CRT-exponent d_p is typically of size at most 110. We obtain better results in our experiments as shown in Table 4.

5 The Third Type of Equations

In this section, we give our main algorithm to find small roots of extended simultaneous modular univariate linear equations. At first, we introduce this kind of equations.

Extended Simultaneous Modular Univariate Linear Equations. Given positive integers $r, r_1, \ldots, r_n$ and $N, a_1, \ldots, a_n$ and bounds $\gamma_1, \ldots, \gamma_n, \eta \in (0,1)$. Suppose that $N = 0 \bmod p^r$ and $p \geq N^\eta$. We want to find all integers $(x_1^{(0)}, \ldots, x_n^{(0)})$ such that $|x_1^{(0)}| \leq N^{\gamma_1}, \ldots, |x_n^{(0)}| \leq N^{\gamma_n}$, and

$$\begin{cases} f_1(x_1^{(0)}) = a_1 + x_1^{(0)} = 0 \bmod p^{r_1} \\ f_2(x_2^{(0)}) = a_2 + x_2^{(0)} = 0 \bmod p^{r_2} \\ \qquad\qquad \vdots \\ f_n(x_n^{(0)}) = a_n + x_n^{(0)} = 0 \bmod p^{r_n} \end{cases}$$

5.1 Our Main Result

Our main result is as follows:

Theorem 10. *Under Assumption 1, the above equations can be solved provided that*

$$\sqrt[n]{\frac{\gamma_1 \cdots \gamma_n}{r r_1 \cdots r_n}} < \eta^{\frac{n+1}{n}} \quad \text{and} \quad \eta \gg \frac{1}{\sqrt{\log N}}$$

The running time of the algorithm is polynomial in $\log N$ *but exponential in* n.

Proof. First, for every j ($j \in \{1, \ldots, n\}$), we check whether condition $\frac{\gamma_j}{r_j} \leq \eta$ is met. If there exists k such that $\frac{\gamma_k}{r_k} > \eta$, then we throw away this corresponding polynomial $f_k(x)$, since this polynomial could not offer any useful information. Here suppose that all the polynomials satisfy our criteria. Define a collection of polynomials as follows:

$$f_{[i_1,\ldots,i_n]}(x_1, \ldots, x_n) = (a_1 + x_1)^{i_1} \cdots (a_n + x_n)^{i_n} N^{\max\{\lceil \frac{t - \sum_{j=1}^{n} r_j i_j}{r} \rceil, 0\}}$$

Notice that for all indexes $i_1, \ldots, i_n$, $f_{[i_1,\ldots,i_n]}(x_1^{(0)}, \ldots, x_n^{(0)}) = 0 \bmod p^t$. We select the collection of shift polynomials that satisfies

$$0 \leq \sum_{j=1}^{n} \gamma_j i_j \leq \eta t$$

The reason we select these shift polynomials is that we try to select as many helpful polynomials as possible by taking into account the sizes of the root bounds.

We define the polynomial order $\prec$ as $x_i^{i_1} x_2^{i_2} \cdots x_n^{i_n} \prec x_1^{i_1'} x_2^{i_2'} \cdots x_n^{i_n'}$ if

$$\sum_{j=1}^{n} i_j < \sum_{j=1}^{n} i_j' \quad \text{or} \quad \sum_{j=1}^{n} i_j = \sum_{j=1}^{n} i_j', \quad i_j = i_j' (j = 1, \ldots, k), \quad i_{k+1} < i_{k+1}'$$

Ordered in this way, the basis matrices become triangular in general.

We compute the dimension of lattice $\mathcal{L}$ as w where

$$w = \dim(\mathcal{L}) = \sum_{0 \leq \gamma_i i_1 + \cdots + \gamma_n i_n \leq \beta t} 1 = \frac{(\eta t)^n}{n!} \frac{1}{\gamma_1 \cdots \gamma_n} + o(t^n)$$

and the determinate $\det(\mathcal{L}) = N^{s_N} X_1^{s_{X_1}} \cdots X_n^{s_{X_n}}$ where

$$s_N = \sum_{0 \leq r_1 i_1 + \cdots + r_n i_n \leq t} \lceil \frac{t - \sum_{j=1}^{n} r_j i_j}{r} \rceil = \frac{t^{n+1}}{(n+1)!} \frac{1}{r r_1 \cdots r_n} + o(t^{n+1})$$

$$s_{X_j} = \sum_{0 \leq \gamma_1 i_1 + \cdots + \gamma_n i_n \leq \eta t} i_j = \frac{t^{n+1}}{(n+1)!} \frac{1}{\gamma_1 \cdots \gamma_{j-1} \gamma_j^2 \gamma_{j+1} \cdots \gamma_n} + o(t^{n+1})$$

for each $s_{X_1}, s_{X_2}, \ldots, s_{X_n}$.

To obtain the number of n polynomials with short coefficients that contain all small roots over integer, we apply LLL basis reduction algorithm to the lattice $\mathcal{L}$. Lemma 1 gives us an upper bound on the norm of the shortest vector in the LLL-reduced basis; if the bound is smaller than the bound given in Lemma 2, we can obtain the desired polynomial. We require the following condition:

$$2^{\frac{w-1}{4}} \det(\mathcal{L})^{\frac{1}{w}} < \frac{N^{\eta t}}{\sqrt{w}} \tag{4}$$

Ignoring low order terms of m and the quantities that do not depend on N, we have the following result

$$s_N + \sum_{j=1}^{n} \gamma_j s_{X_j} < w\eta t$$

After some calculations, we can get the final result

$$\sqrt[n]{\frac{\gamma_1 \cdots \gamma_n}{r r_1 \cdots r_n}} < \eta^{\frac{n+1}{n}}$$

In particular, from the equation (4), in order to ignore the quantities that do not depend on N, we must have

$$2^{\frac{w}{4}} \ll N^{\eta t} \quad \text{and} \quad \det(\mathcal{L})^{\frac{1}{w}} < N^{\eta t}$$

and these inequations imply that

$$w \ll 4\eta t \log_2 N \quad \text{and} \quad \frac{s_N}{w} \log_2 N < \eta t \log_2 N$$

Finally we have

$$\frac{1}{4(n+1) r r_1 \cdots r_n} \ll \eta^2 \log_2 N$$

Furthermore, one can check that in order to let the value $2^{w/4}$ become negligible compared with $N^{\eta t}$, we must have

$$\eta^2 \log_2 N \gg 1$$

The running time is dominated by LLL-reduction, therefore, the total running time for this approach is polynomial in $\log N$ but exponential in n. □

Like [4,36], we also consider the generalization to simultaneous linear equations of higher degree.

Extended Simultaneous Modular Univariate Equations. Suppose that $N = 0 \bmod p^r, p \geq N^\eta$, we consider the simultaneous modular univariate equations

$$\begin{cases} h_1(x_1) = x_1^{\delta_1} + a_{\delta_1} x_1^{\delta_1 - 1} + \cdots + a_1 = 0 \bmod p^{r_1} \\ h_2(x_2) = x_1^{\delta_2} + b_{\delta_2} x_1^{\delta_2 - 1} + \cdots + b_1 = 0 \bmod p^{r_2} \\ \qquad\qquad\qquad\qquad \vdots \\ h_n(x_n) = x_1^{\delta_n} + c_{\delta_n} x_1^{\delta_n - 1} + \cdots + c_1 = 0 \bmod p^{r_n} \end{cases}$$

Here each equation $h_j(x_j)$ has one variable and the degree of $h_j(x_j)$ is δ_j. We give the following result.

Theorem 11. *Under Assumption 1, the above generalised problem can be solved provided that*

$$\sqrt[n]{\frac{\delta_1 \gamma_1 \cdots \delta_n \gamma_n}{r r_1 \cdots r_n}} < \eta^{\frac{n+1}{n}} \quad \textit{and} \quad \eta \gg \frac{1}{\sqrt{\log N}}$$

The running time of the algorithm is polynomial in $\log N$ *but exponential in* n.

The proof is very similar to [4,36], we omit it here.

5.2 Common Prime RSA

In [13], Hinek revisited a new variant of RSA, called Common Prime RSA, where the modulus $N = pq$ is chosen such that $p-1$ and $q-1$ have a large common factor. For convenience, we give a brief description on the property of Common Prime RSA. Without loss of generality, assume that $p = 2ga+1$ and $q = 2gb+1$, where $g \simeq N^{\gamma}$ and a, b are coprime integers, namely $\gcd(a, b) = 1$. The decryption exponent d and encryption exponent e satisfy that

$$ed \equiv 1 \bmod 2gab \tag{5}$$

where $e \simeq N^{1-\gamma}$ and $d \simeq N^{\beta}$.

For a better comparison with the previous attacks, we give a brief review on all known attacks.

Wiener's attack [38]. Using a continued fraction attack, Wiener proved that given any valid Common Prime RSA public key (N, e) with private exponent $d < N^{\frac{1}{4}-\frac{\gamma}{2}}$, namely $\beta < \frac{1}{4} - \frac{\gamma}{2}$, one can factor N in polynomial-time.

Hinek's attack [13]. Hinek revisited this problem and proposed two lattice-based attacks. Due to Hinek's work, when $\beta < \gamma^2$ or $\beta < \frac{2}{5}\gamma$, N can be factored in polynomial-time.

Jochemsz-May's attack [17]. Jochemsz and May gave another look at the equation proposed by Hinek [13] and modified the unknown variables in the equation. The bound has been further improved as

$$\beta < \frac{1}{4}(4 + 4\gamma - \sqrt{13 + 20\gamma + 4\gamma^2}).$$

Sarkar-Maitra's attack [33]. Sarkar and Maitra proposed two improved attacks, one attack worked when $\gamma \leq 0.051$, and another worked when $0.051 < \gamma \leq 0.2087$.

One can check that when $\gamma \geq 0.2087$, Jochemsz-May's attack [17] is superior to other attacks. We use the algorithm of Theorem 10 to make an improvement on previous attacks when $\gamma \geq 0.3872$. We give a comparison with Jochemsz-May's attack in Fig. 3.

Our results improve Jochemsz-May's attack dramatically when γ is large, for instance, when γ is close to 0.5, we improve the bound on β from 0.2752, which is the best result of previous attacks, to 0.5. Below is our main result.

Theorem 12. *Assume that there exists instance of Common Prime RSA $N = pq$ with the above-mentioned parameters. Under Assumption 1, N can be factored in polynomial-time provided*

$$\beta < 4\gamma^3 \quad \textit{and} \quad \gamma > \frac{1}{4}$$

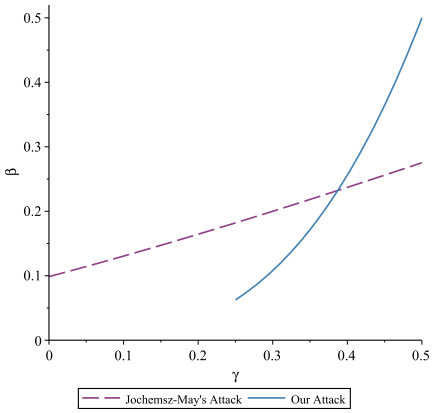

Fig. 3. Comparison of our theoretical bounds with Jochemsz-May's work.

Proof. According to the property of Common Prime RSA, we have $N = pq = (2ga + 1)(2gb + 1)$ which implies $N - 1 \equiv 0 \bmod g$. On the other hand, from Equation (5) one can obtain

$$ed - 1 \equiv 0 \bmod g$$

Multiplying by the inverse of e modulo $N - 1$, we can obtain the following equation,

$$E - x \equiv 0 \bmod g$$

where E denotes the inverse of e modulo $N - 1$ and x denotes the unknown d. Moreover, since $(p - 1)(q - 1) = 4g^2ab$, we have another equation,

$$N - y \equiv 0 \bmod g^2$$

where y denotes the unknown $p + q - 1$.

In summary, simultaneous modular univariate linear equations can be listed as

$$\begin{cases} E - x \equiv 0 \bmod g \\ N - y \equiv 0 \bmod g^2 \end{cases}$$

Note that $N - 1$ is a multiple of g and $(d, p + q - 1)$ is the desired solution of above equations, where $g \simeq N^{\gamma}$, $d \simeq N^{\beta}$ and $p + q - 1 \simeq N^{\frac{1}{2}}$. Obviously, this kind of modular equations is what we considered in Theorem 10. Setting

$$n = 2,\ r = 1,\ r_1 = 1,\ r_2 = 2,\ \gamma_1 = \beta,\ \gamma_2 = \frac{1}{2},\ \eta = \gamma$$

We have

$$\gamma > \beta \quad \gamma > \frac{1}{4} \quad \beta < 4\gamma^3$$

Table 5. Comparison of our theoretical and experimental results with existing works.

γ	Theo. of [17]	Our result			
		Theo.	Expt.	Dim	Time (in seconds)
0.40	0.237	0.256	0.220	86	12321.521
0.42	0.245	0.294	0.260	113	53669.866
0.45	0.256	0.354	0.320	105	29128.554
0.48	0.268	0.415	0.390	98	15058.558

Then we can obtain

$$\beta < 4\gamma^3 \quad \text{and} \quad \gamma > \frac{1}{4}$$

Under Assumption 1, one can solve the desired solution. This concludes the proof of Theorem 12. □

Experimental Results. Some experimental data on the different size of g are listed in Table 5. Here we used 1000-bit N. Assumption 1 worked perfectly in all the cases. We always succeed to find out our desired roots.

6 Conclusion

In this paper, we consider three type of generalized equations and propose some new techniques to find small root of these equations. Applying our algorithms, we obtain the best analytical/experimental results for some attacks on RSA and its variants. Besides, we believe that our new algorithms may find new applications in various other contexts.

Acknowledgments

We would like to thank the anonymous reviewers for helpful comments. This research was supported by CREST, JST. Part of this work was also supported by Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA06010703, No. XDA06010701 and No. XDA06010702), the National Key Basic Research Project of China (No. 2011CB302400 and No. 2013CB834203), and National Science Foundation of China (No. 61379139 and No. 61472417).

References

1. D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Transactions on Information Theory, 46(4):1339–1349, 2000.
2. D. Boneh, G. Durfee, and N. Howgrave-Graham. Factoring $N = p^r q$ for large r. CRYPTO 1999, LNCS, vol.1666, pages 787–787. Springer, 1999.

3. G. Castagnos, A. Joux, F. Laguillaumie, and P. Nguyen. Factoring pq^2 with quadratic forms: Nice cryptanalyses. ASIACRYPT 2009, LNCS, vol.5912, pages 469–486. Springer, 2009.
4. H. Cohn and N. Heninger. Approximate common divisors via lattices. ANTS-X, 2012.
5. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology, 10(4):233–260, 1997.
6. J.S. Coron, A. Joux, I. Kizhvatov, D. Naccache, and P. Paillier. Fault attacks on RSA signatures with partially unknown messages. CHES 2009, LNCS, vol.5747, pages 444–456. Springer, 2009.
7. J.S. Coron, D. Naccache, and M. Tibouchi. Fault attacks against EMV signatures. CT-RSA 2010, LNCS, vol.5985, pages 208–220. Springer, 2010.
8. The EPOC and the ESIGN Algorithms. IEEE P1363: Protocols from Other Families of Public-Key Algorithms, 1998.
   http://grouper.ieee.org/groups/1363/StudyGroup/NewFam.html
9. M. Ernst, E. Jochemsz, A. May, and B. De Weger. Partial key exposure attacks on RSA up to full size exponents. EUROCRYPT 2005, LNCS, vol.3494, pages 555–555. Springer, 2005.
10. P.A. Fouque, N. Guillermin, D. Leresteux, M. Tibouchi, and J.C. Zapalowicz. Attacking RSA–CRT signatures with faults on montgomery multiplication. Journal of Cryptographic Engineering, 3(1):59–72. Springer, 2013.
11. M. Herrmann. Improved cryptanalysis of the multi-prime Φ-hiding assumption. AFRICACRYPT 2011, LNCS, vol.6737, pages 92–99. Springer, 2011.
12. M. Herrmann and A. May. Solving linear equations modulo divisors: On factoring given any bits. ASIACRYPT 2008, LNCS, vol.5350, pages 406–424. Springer, 2008.
13. M. Hinek. Another look at small RSA exponents. CT-RSA 2006, LNCS, vol.3860, pages 82–98. Springer, 2006.
14. N. Howgrave-Graham. Finding small roots of univariate modular equations revisited. IMACC 1997, LNCS, vol.1355, pages 131–142. Springer, 1997.
15. N. Howgrave-Graham. Approximate integer common divisors. Cryptography and Lattices, LNCS, vol.2146, pages 51–66. Springer, 2001.
16. K. Itoh, N. Kunihiro, and K. Kurosawa. Small secret key attack on a variant of RSA (due to Takagi). CT-RSA 2008, LNCS, vol.4964, pages 387–406. Springer, 2008.
17. E. Jochemsz and A. May. A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. ASIACRYPT 2006, LNCS, vol.4284, pages 267–282. Springer, 2006.
18. A. Joux. Algorithmic cryptanalysis. Chapman & Hall/CRC, 2009.
19. K. Tosu and N. Kunihiro. Optimal bounds for multi-prime Φ-hiding assumption. ACISP 2012, LNCS, vol.4450, pages 55–69. Springer, 2012.
20. A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. Mathematische Annalen, 261(4):515–534, 1982.
21. A. May. New RSA vulnerabilities using lattice reduction methods. PhD thesis, 2003.
22. A. May. Secret exponent attacks on RSA-type schemes with moduli $N = p^r q$. PKC 2004, LNCS, vol.2947, pages 218–230. Springer, 2004.
23. A. May. Using LLL-reduction for solving RSA and factorization problems. The LLL algorithm, pages 315–348, 2010.
24. A. May and M. Ritzenhofen. Implicit factoring: On polynomial time factoring given only an implicit hint. PKC 2009, LNCS, vol.5443, pages 1–14. Springer, 2009.

25. P. Nguyen and D. Stehlé. Floating-Point LLL revisited. EUROCRYPT 2005, LNCS, vol.3494, pages 215–233. Springer, 2005.
26. A. Nitaj. A new attack on RSA and CRT-RSA. AFRICACRYPT 2012, LNCS, vol.7374, pages 221–233. Springer, 2012.
27. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. Eurocrypt 1998, LNCS, vol.1403, pages 308–318. Springer, 1998.
28. R. Rivest and A. Shamir. Efficient factoring based on partial information. EUROCRYPT 1985, LNCS, vol.219, pages 31–34. Springer, 1986.
29. S. Sarkar. Reduction in lossiness of RSA trapdoor permutation. SPACE 2012, LNCS, vol.7644, pages 144–152. Springer, 2012.
30. S. Sarkar. Revisiting prime power RSA. Cryptology ePrint Archive, Report 2015/774, 2015. http://eprint.iacr.org/
31. S. Sarkar. Small secret exponent attack on RSA variant with modulus $N = p^r q$. Designs, Codes and Cryptography, pages 1–10, 2014.
32. S. Sarkar and S. Maitra. Approximate integer common divisor problem relates to implicit factorization. IEEE Transactions on Information Theory, 57(6):4002–4013, 2011.
33. S. Sarkar and S. Maitra. Cryptanalytic results on Dual CRT and Common Prime RSA. Designs, Codes and Cryptography, 66(1-3):157–174, 2013.
34. A. Shamir. A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem. FOCS 1982, pages 145–152. IEEE, 1982.
35. T. Takagi. Fast RSA-type cryptosystem modulo $p^k q$. Crypto 1998, LNCS, vol.1462, pages 318–326. Springer, 1998.
36. A. Takayasu and N. Kunihiro. Better lattice constructions for solving multivariate linear equations modulo unknown divisors. ACISP 2013, LNCS, vol.7959, pages 118–135. Springer, 2013.
37. M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. EUROCRYPT 2010, LNCS, vol.6110, pages 24–43. Springer, 2010.
38. M.J. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory, 36(3):553–558, 1990.