Partial Key Exposure Attacks on Takagi's Variant of RSA

Zhangjie Huang[1,2,3], Lei Hu[1,2], Jun Xu[1,2],
Liqiang Peng[1,2], and Yonghong Xie[1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[2] Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing 100093, China
[3] University of Chinese Academy of Sciences, Beijing 100049, China
{zhjhuang,hu,xjun,lqpeng,yhxie}@is.ac.cn

Abstract. We present several attacks on a variant of RSA due to Takagi when different parts of the private exponent are known to an attacker. We consider three cases when the exposed bits are the most significant bits, the least significant bits and the middle bits of the private exponent respectively. Our approaches are based on Coppersmith's method for finding small roots of modular polynomial equations. Our results extend the results of partial key exposure attacks on RSA of Ernst, Jochemsz, May and Weger (EUROCRYPT 2005) for moduli from $N = pq$ to $N = p^r q$ ($r \geq 2$).

Keywords: RSA, partial key exposure, Coppersmith's method, lattice reduction, LLL algorithm.

1 Introduction

In his seminal work [5] in 1996, Coppersmith described a method for finding small roots of univariate modular polynomial equations in polynomial time based on lattice basis reduction. Coppersmith showed that for a monic univariate polynomial $f(x)$ of degree d, one can find any root x_0 of $f(x) \equiv 0 \pmod{N}$ in polynomial time if $|x_0| < N^{1/d}$. The essence of Coppersmith's method is to find integral linear combinations of polynomials which share a common root modulo some integer such that the result has small coefficients. Thus one may obtain a polynomial with the desired root over the integers and one can then find the desired root using standard root-finding algorithms. This method was then reformulated by Howgrave-Graham [11] in a simpler way which has been widely adopted by researchers for cryptanalysis. In general, the reformulation by Howgrave-Graham is used when we say Coppersmith's method.

Coppersmith's method can be extended to handle multivariate modular polynomial equations with some heuristic assumptions. In the multivariate cases, we obtain some multivariate integer polynomials and find the final roots by computing the resultants or using Gröbner basis algorithms. At present, there have

I. Boureanu, P. Owesarski, and S. Vaudenay (Eds.): ACNS 2014, LNCS 8479, pp. 134–150, 2014.

been many variants of Coppersmith's method. In 2006, Jochemsz and May [13] described a general strategy for finding small roots of modular or integer multivariate polynomial equations. Their strategy makes it easier to construct lattices and to analyse the bounds for the small roots. More recently, Herrmann and May [9] introduced the technique of unravelled linearization and Aono [1] introduced the Minkowski sum based lattice construction. All these variants make Coppersmith's method a powerful tool in the field of cryptanalysis.

Since the invention of Coppersmith's method, much effort has been made to evaluate the security of RSA and its variants. It was used to break RSA when the private exponent $d < N^{0.292}$ [3] and to attack CRT-RSA when the private exponent is small [14]. It was also used to prove the equivalence between knowing the private exponent d and factoring the modulus N [6,15]. The book [10] is a good survey of these kinds of applications.

In order to gain a faster decryption, Takagi [18] proposed a RSA-type cryptosystem with moduli $N = p^r q$. The polynomial-time equivalence between factoring the modulus and recovering the private exponent for Takagi's scheme was proved in [15]. Later in [12], Itoh, Kunihiro and Kurosawa extended the method of lattice construction in [15] and gave a polynomial-time attack when the private exponent $d < N^{\frac{7-2\sqrt{7}}{3(r+1)}}$ (and improved to $d < N^{\frac{2-\sqrt{2}}{r+1}}$ by using "Geometrically Progressive Matrices"). Both in [15] and [12], the authors constructed lattices in a clever way by taking advantage of the foreknowledge that $y^r z = N$ where variables y, z denote p, q respectively. They substituted N for every occurrence of $y^r z$ while constructing lattices. This kind of foreknowledge was not used in Jochemsz and May's strategy [13]. This trick of substituting was first used in [3] and then also adopted in [7].

In this paper, we consider the partial key exposure attacks on Takagi's variant of RSA. The partial key exposure attacks were first considered by Boneh, Durfee and Frankel in [4]. The work was then followed by Blömer and May in [2] and Ernst *et al.* in [8]. The first attack we present in this paper is for the case when some of the most significant bits (MSBs) of the private exponent are known. Our attack extends the method of constructing lattices in [12]. Our second attack on knowing some of the least significant bits (LSBs) of the private exponent can be achieved in an analogous way as it was done in [12]. We also consider how to attack the case when the known bits lie in the middle of the private exponent. All our attacks are based on Coppersmith's method. We summarize our results in the following theorems and prove them in Section 3.

Our attack results on known MSBs and on known bits in the middle of the private exponent are asymptotically the same, *i.e.*, the two attacks need the same amount of known bits. We state the results of these two attacks in Theorem 1:

Theorem 1 (Known MSBs/Known Bits in the Middle). *For any $\epsilon > 0$ there exists N_0 such that if $N > N_0$ for $N = p^r q$ where p and q are primes with the same bit-length, the following holds: Let $e = N^{\alpha}$ and $d = N^{\beta}$ be integers satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$ and $\gcd(e, p) = 1$. Given about $(1 - \frac{\delta}{\beta})$-fraction of the MSBs or continuous bits in the middle of d, the modulus N can be factored in polynomial time if*

$$\delta \leq \frac{7}{4(r+1)} - \frac{1}{4}\sqrt{\frac{24(\alpha+\beta)}{r+1} - \frac{39}{(r+1)^2}} - \epsilon.$$

We show our result of Theorem 1 in Fig. 1 with $r = 2$. The figure shows the relation between the fraction of bits required for an attack and the size of d when we set e as full-size, *i.e.*, $\alpha = 2/(r+1)$. The left rectangle in the figure represents the result of the small key attack from [12].

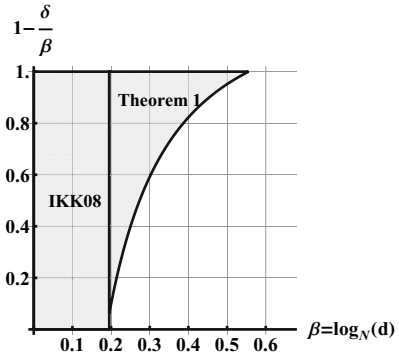

Fig. 1. Known MSBs/Known bits in the middle attack: The relation between the fraction of bits required and the size of d when $r = 2$ and $\alpha = 2/(r+1)$

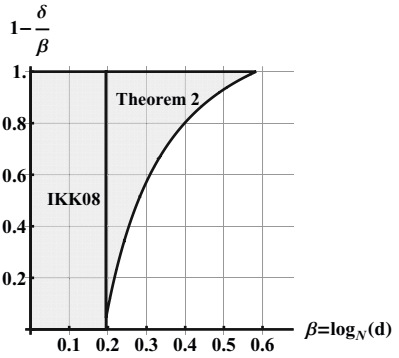

Fig. 2. Known LSBs attack: The relation between the fraction of bits required and the size of d when $r = 2$ and $\alpha = 2/(r+1)$

We note that when $r = 1$ and $\alpha = 1$, our result on known MSBs is a little bit weaker than one of the results in [8] (Section 4.1.1 therein):

$$\delta < \frac{5}{6} - \frac{1}{3}\sqrt{1+6\beta}.$$

In this case, the way of constructing lattices in [8] is better than ours because our method is not able to make the best of the information we get. We treat an equation which is actually over the integers as a modular equation (see Section 3.1). On the other hand, our result is a general result when $r \geq 1$ in $N = p^r q$. The case of known MSBs can be viewed as a special case of known bits in the middle.

Theorem 2 (Known LSBs). *For any $\epsilon > 0$ there exists N_0 such that if $N > N_0$ for $N = p^r q$ where p and q are primes with the same bit-length, the following holds: Let $e = N^\alpha$ and $d = N^\beta$ be integers satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$ and $\gcd(e, p) = 1$. Given about $(1 - \frac{\delta}{\beta})$-fraction of the LSBs of d, the modulus N can be factored in polynomial time if*

$$\delta \leq \frac{5}{3(r+1)} - \frac{2}{3}\sqrt{\frac{3(\alpha+\beta)}{r+1} - \frac{5}{(r+1)^2}} - \epsilon.$$

Fig. 2 illustrates our result when the least significant bits of d are known. We set $r = 2$ and $\alpha = 2/(r+1)$ in Fig. 2. The figure shows the relation between the fraction of bits required for an attack and the size of d. The left rectangle in the figure represents the result of the small key attack from [12]. Our result when $r = 1$ is the same with the result in [8]. Our result may be seen as an extension of the result in [8] for moduli $N = p^r q$ when $r \geq 2$.

Our results stated above are general results for exponents (e, d) with arbitrary sizes. From the bounds for δ in these two theorems, the relations between the fraction of bits required and the size of e when d is full-size are also clear, we omit the corresponding figures here. One may notice that the size of e (represented as α) and the size of d (represented as β) have the same impact on the attacks. Intuitively, the quality of our attacks depends on the information we know, including the public exponent e and the known bits of d (and others). There is a trade-off between the size of e and the size of known bits of d. We can mount the attacks in the cases when e is smaller and d is larger (which means that we know more bits of d) and vice versa, as long as we know approximately the same number of bits. From this point of view, our results are reasonable intuitively.

The rest of this paper is organized as follows. Section 2 gives some preliminaries on lattices and also a brief description of Takagi's variant of RSA. We derive our problems from Takagi's variant of RSA in Section 3 and give our approaches to the problems. The justification of our approaches is also examined through some experiments in Section 4. Finally, we give our conclusion in Section 5.

2 Preliminaries

Coppersmith's method uses lattice basis reduction to find the polynomials with small coefficients. Hence we briefly introduce a few necessary definitions and facts about lattices. It is common to use the LLL algorithm along with Howgrave-Graham's lemma to estimate the bounds for the small roots. This was stated in Howgrave-Graham's reformulation [11] of Coppersmith's method. Finally we introduce Takagi's variant of RSA.

2.1 Lattices and Howgrave-Graham's Lemma

Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_\omega \in \mathbb{Z}^n$ be linearly independent (row) vectors. A lattice L generated by $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_\omega$ is the set of all integral linear combinations of these vectors:

$$L = \mathcal{L}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_\omega) = \left\{ \boldsymbol{v} \in \mathbb{Z}^n \mid \boldsymbol{v} = \sum_{i=1}^{\omega} a_i \boldsymbol{b}_i,\ a_i \in \mathbb{Z} \right\}.$$

We call n the dimension of L and ω its rank. We often denote the basis $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_\omega$ as a matrix, called the basis matrix of L:

$$B = \begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_\omega \end{pmatrix} \in \mathbb{Z}^{\omega \times n}.$$

Then the determinant of L can be computed as $\det(L) = \sqrt{\det(BB^T)}$.

The most famous algorithm for lattice basis reduction is the LLL algorithm [16]. It allows one to find a short vector in a lattice in polynomial time. The proof of the following fact can be found in [17].

Fact 1 (LLL). *Let L be a lattice spanned by the rows of $B = (\boldsymbol{b}_1^T, \ldots, \boldsymbol{b}_\omega^T)^T$. The LLL algorithm outputs a reduced basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\omega$ satisfying*

$$\|\boldsymbol{v}_i\| \leq 2^{\frac{\omega(\omega-1)}{4(\omega-i+1)}} \det(L)^{\frac{1}{\omega-i+1}},\ 1 \leq i \leq \omega$$

in polynomial time in ω and in the bit size of the entries of the basis matrix B.

When using Coppersmith's method to find the small roots of a modular polynomial equation, the following lemma due to Howgrave-Graham is useful. It states that under which condition a modular equation holds over the integers. The norm of a polynomial $f(x_1, \ldots, x_n) = \sum a_{i_1,\ldots,i_n} x_1^{i_1} \ldots x_n^{i_n}$ is defined as $\|f(x_1, \ldots, x_n)\| = \sqrt{\sum |a_{i_1,\ldots,i_n}|^2}$.

Lemma 1 (Howgrave-Graham [11]). *Let $g(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ be a polynomial that consists of at most ω monomials. Suppose that*

1. *$g(x_1^{(0)}, \ldots, x_n^{(0)}) \equiv 0 \pmod{N}$ for $|x_1^{(0)}| \leq X_1, \ldots, |x_n^{(0)}| \leq X_n$, and*
2. *$\|g(X_1 x_1, \ldots, X_n x_n)\| < \frac{N}{\sqrt{\omega}}$,*

then $g(x_1^{(0)}, \ldots, x_n^{(0)}) = 0$ holds over the integers.

Combining the Howgrave-Graham's lemma with the LLL algorithm, we deduce that if

$$2^{\frac{\omega(\omega-1)}{4(\omega-i+1)}} \det(L)^{\frac{1}{\omega-i+1}} < \frac{N}{\sqrt{\omega}},$$

then the polynomials corresponding to the shortest i reduced basis vectors satisfy Howgrave-Graham's bound. The condition implies

$$\det(L) < 2^{-\frac{\omega(\omega-1)}{4}} (\frac{1}{\sqrt{\omega}})^{\omega-i+1} N^{\omega-i+1}.$$

As in previous works, we ignore the terms that do not depend on N and simply check the condition $\det(L) < N^{\omega-i+1}$. In practice, this is convenient when N is large enough. After obtaining enough equations over the integers, one can extract the common roots by computing the resultants of these polynomials under the following heuristic assumption:

Assumption 1. *The resultant computations for the polynomials corresponding to the first few LLL-reduced basis vectors produce non-zero polynomials.*

The above assumption may sometimes fail especially for the cases dealing with four or more variables. If this assumption fails, we may obtain the roots in other ways (See the note in Section 4.).

2.2 Takagi's RSA-Type Cryptosystem

In 1998, Takagi [18] proposed a cryptosystem with moduli $N = p^r q$ based on RSA aiming at a faster decryption process and keeping its security at the same time. We give a brief description of Takagi's cryptosystem here.

Generate two primes p and q with the same bit-length and let $N = p^r q$ for some small integer $r \geq 2$. Let e and d be integers satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$ and $\gcd(e, p) = 1$. Then set (N, e) as the public key and (p, q, d) as the private key. The encryption of a message $M \in \mathbb{Z}_N^*$ is like in the RSA cryptosystem: $C = M^e \bmod N$. The decryption process is as follows. Firstly, compute $M_p = C^d \bmod p$ and $M_q = C^d \bmod q$. It is clear that $M \equiv M_q \pmod{q}$. Then compute an integer M_{p^r} satisfying $M \equiv M_{p^r} \pmod{p^r}$ from M_p and C using Hensel lifting. At last M is obtained from M_q and M_{p^r} using the Chinese Remainder Theorem. We refer to the original article [18] for more details especially for the Hensel lifting computation.

3 Description of Attacks and Proof of Theorems

In this section, we give our attack methods to the problems when one of the three different parts of the private exponent d in Takagi's variant of RSA is known. Here the three different parts are the most significant bits, the least significant bits and the continuous bits lying in the middle of d. We first describe how to construct lattices from the problems and then prove the results stated in Section 1. The first two types of exposure bits are already considered in [8] and we treat them in an analogous way in this paper. We consider a new type of known bits, *i.e.*, bits in the middle of d. Our methods extend the way of constructing lattices in [12].

3.1 Attack with Known MSBs

Deriving the problem. In Takagi's variant of RSA, we have

$$N = p^r q \text{ and } ed \equiv 1 \bmod (p-1)(q-1),$$

where p and q are two primes with the same bit-length. There exists an integer k such that

$$ed = 1 + k(p-1)(q-1). \tag{1}$$

We assume the exponents $e = N^\alpha$ and $d = N^\beta$. Since the bit-lengths of p and q are the same, then it holds that $p < 2N^{1/(r+1)}$ and $q < 2N^{1/(r+1)}$. Since $e < (p-1)(q-1)$ and $d < (p-1)(q-1)$, we have $0 < \alpha, \beta < 2/(r+1)$. According to (1), we have

$$k = \frac{ed-1}{(p-1)(q-1)} < \frac{2ed}{pq} < 2N^{\alpha+\beta-\frac{2}{r+1}}.$$

If we know some of the most significant bits of the private exponent d, that is we know $\tilde{d}$ such that $d = \tilde{d} + d_0$, where d_0 is the unknown part of d satisfies $|d_0| = |d - \tilde{d}| < N^\delta$ for $\delta < \beta$. Then we can rewrite (1) as

$$e(\tilde{d} + d_0) = 1 + k(p-1)(q-1),$$

and we know that the polynomial

$$f_{\mathrm{msb}}(w, x, y, z) = x(y-1)(z-1) + ew + 1$$

has a root $(-d_0, k, p, q)$ modulo $N_0 = e\tilde{d}$ ($\approx N^{\alpha+\beta}$). Note that we view an integer equation as a modular equation here. Define

$$W = N^\delta,\ X = 2N^{\alpha+\beta-2/(r+1)} \text{ and } Y = Z = 2N^{1/(r+1)}.$$

We are trying to find a "small" root $(w_0, x_0, y_0, z_0) = (-d_0, k, p, q)$ of

$$f_{\mathrm{msb}}(w, x, y, z) \equiv 0 \pmod{N_0}$$

with the bounds $|w_0| < W$, $|x_0| < X$, $|y_0| < Y$ and $|z_0| < Z$.

Constructing the Lattice Basis. The first step of our attack is to collect some polynomials which share a common root (w_0, x_0, y_0, z_0) modulo N_0^m for some fixed positive integer m which is polynomial in $\frac{1}{\epsilon}$. We define these polynomials as (the form)

$$g_{i_1,i_2,i_3,i_4,i_5}(w, x, y, z) = w^{i_1} x^{i_2} y^{i_3} z^{i_4} f_{\mathrm{msb}}(w, x, y, z)^{i_5} N_0^{m-i_5},\ \text{for } 0 \le i_5 \le m,$$

where the indices $(i_1, i_2, i_3, i_4, i_5)$ will be determined later. We then construct a lattice basis with the coefficient vectors of $g_{i_1,i_2,i_3,i_4,i_5}(wW, xX, yY, zZ)$ as its basis vectors. The principle of the choice for $(i_1, i_2, i_3, i_4, i_5)$ is that we collect an (ordered) list of polynomials $G = \{\, g_{i_1,i_2,i_3,i_4,i_5} \,\}$ such that every polynomial in the list introduces exactly one monomial that does not appear in the previous polynomials. This will make the basis triangular which allows for an easy determinant calculation.

Since our choice for the polynomials is based on the polynomials Itoh, Kunihiro and Kurosawa chose in [12], we briefly introduce the construction of basis in [12]. In [12], they considered the problem of finding the small root (k, p, q) of the modular polynomial

$$\bar{f}(x, y, z) = x(y-1)(z-1) + 1 \bmod e, \tag{2}$$

where e, k, p and q are the same meanings as in (1). They constructed a basis which was triangular by taking advantage of the relation that $y^r z = N$ since $N = p^r q$ is public. Every occurrence of $y^r z$ in the polynomials was replaced

Algorithm 1. The way of collecting polynomials in [12] for integers n and s

$G_n \leftarrow \emptyset$
for $u = 0, \ldots, n$ **do**
  for $i = 0, \ldots, u-1$ **do**
    append $\bar{g}_{u-i,0,0,i},\ \bar{g}_{u-i,1,0,i}$ to G_n
    append $\bar{g}_{u-i,r-1,1,i},\ \bar{g}_{u-i,r-2,1,i}, \ldots,\ \bar{g}_{u-i,1,1,i}$ to G_n
  for $j = 0, \ldots, s$ **do**
    append $\bar{g}_{0,j,0,u}$ to G_n
  for $k = 1, \ldots, s$ **do**
    append $\bar{g}_{0,r-1,k,u},\ \bar{g}_{0,r-2,k,u}, \ldots,\ \bar{g}_{0,0,k,u}$ to G_n
return G_n

by N and thus changed the monomials in the polynomials. All our settings are the same with theirs except that we consider the polynomial

$$f_{\mathrm{msb}}(w, x, y, z) = \bar{f}(x, y, z) + ew \bmod N_0 \tag{3}$$

instead of $\bar{f}(x, y, z)$.

For a fixed positive integer n and some integer s, they defined the list of polynomials, which we denote as G_n here, according to Algorithm 1. In Algorithm 1, $\bar{g}_{j_1,j_2,j_3,j_4} = x^{j_1} y^{j_2} z^{j_3} \bar{f}^{j_4} e^{n-j_4}$, which means that all polynomials in G_n satisfy $\bar{g}_{j_1,j_2,j_3,j_4}(k, p, q) = 0 \bmod e^n$. Obviously, with the same s, $G_n \subset G_{n+1}$ for any $n \geq 0$.

Now we come to our construction. The idea behind our choice for the polynomials is as follows. First we will show that how to order the monomials can we obtain a basis which is triangular. For any positive integer a, we have the binomial expansion

$$f_{\mathrm{msb}}^a = (\bar{f} + ew)^a = \underbrace{(ew)^a}_{w^a} + \underbrace{\binom{a}{1}(ew)^{a-1}\bar{f}}_{w^{a-1}} + \underbrace{\binom{a}{2}(ew)^{a-2}\bar{f}^2}_{w^{a-2}} + \cdots + \underbrace{\bar{f}^a}_{w^0}. \tag{4}$$

We partition the set of monomials in f_{msb}^a into $a+1$ subsets naturally in terms of the exponent of w in the monomials in (4). Therefore, we order all the monomials in f_{msb}^a in the lattice basis by this sequence: the monomials in the term w^a, then the monomials in the term $w^{a-1}\bar{f}$, and so on.

For $0 \leq b \leq a$, we know from [12] that we can construct a triangular basis from the polynomials in G_b (as in Algorithm 1 when $n = b$) for the monomials in $\bar{f}^b$. Obviously we can construct a triangular basis from the polynomials in $w^{a-b}G_b$ for the monomials in $w^{a-b}\bar{f}^b$. As an abuse of notation, we denote $w^{a-b}G_b$ as the set of polynomials in G_b each multiplied by the term w^{a-b}. We use similar notations hereafter. We then concatenate all the triangular basis for $0 \leq b \leq a$ and end up with a triangular basis for the monomials in f_{msb}^a.

We summarize this process in Algorithm 2. In Algorithm 2, we fix integers m and s and define the list of polynomials G we chose for our problem as follows:

$$G = \bigcup_{i=0}^{m} w^{m-i} G_i = w^m G_0 \bigcup w^{m-1} G_1 \bigcup w^{m-2} G_2 \bigcup \cdots \bigcup w^0 G_m. \tag{5}$$

Algorithm 2. Collecting the polynomials and the corresponding monomials

$G \leftarrow \emptyset,\ H \leftarrow \emptyset$
for $v = m, \ldots, 0$ **do**
  for $u = 0, \ldots, m - v$ **do**
    for $i = 0, \ldots, u - 1$ **do**
      append $g_{v,u-i,0,0,i},\ g_{v,u-i,1,0,i}$ to G
      append $w^v x^u z^i,\ w^v x^u y^{i+1}$ to H
      append $g_{v,u-i,r-1,1,i},\ g_{v,u-i,r-2,1,i}, \ldots,\ g_{v,u-i,1,1,i}$ to G
      append $w^v x^u y^{r-1} z^{i+1},\ w^v x^u y^{r-2} z^{i+1}, \ldots,\ w^v x^u y z^{i+1}$ to H
    for $j = 0, \ldots, s$ **do**
      append $g_{v,0,j,0,u}$ to G
      if $j = 0$ **then**
        append $w^v x^u z^u$ to H
      else
        append $w^v x^u y^{u+j}$ to H
    for $k = 1, \ldots, s$ **do**
      append $g_{v,0,r-1,k,u},\ g_{v,0,r-2,k,u}, \ldots,\ g_{v,0,0,k,u}$ to G
      append $w^v x^u y^{r-1} z^{u+k},\ w^v x^u y^{r-2} z^{u+k}, \ldots,\ w^v x^u z^{u+k}$ to H
return $G,\ H$

We also denote the list of corresponding monomials introduced by the polynomials in G as H.

Remark 1. We must stress that the G_i in (5) is not totally the same as the ones that Algorithm 1 output. We write (5) for ease of presentation. They differ in two places and the purpose of these two replacements is to make sure that the polynomials we chose in G satisfy $g_{i_1,i_2,i_3,i_4,i_5}(w_0, x_0, y_0, z_0) \equiv 0 \bmod N_0^m$.

1. The factor e is replaced by N_0. This replacement does not influence the structure of polynomials, *i.e.*, the monomials they contain;
2. $\bar{f}$ is replaced by f_{msb}. We will show that this replacement does not affect the property of triangular of the final basis. For some $0 \le i \le m$, we consider the polynomials in $w^{m-i}G_i$. We know from Algorithm 1 that each polynomial $\bar{g}$ in G_i is in the form of $\bar{g} = x^{j_1} y^{j_2} z^{j_3} \bar{f}^{j_4} e^{i-j_4}$ with $j_4 \le i$. After the replacements of e to N_0 and $\bar{f}$ to f_{msb}, the corresponding polynomial in $w^{m-i}G_i$ becomes $w^{m-i} x^{j_1} y^{j_2} z^{j_3} f_{\mathrm{msb}}^{j_4} N_0^{i-j_4}$. Rewrite this as

$$
\begin{aligned}
& w^{m-i} x^{j_1} y^{j_2} z^{j_3} f_{\mathrm{msb}}^{j_4} N_0^{i-j_4} \\
=& w^{m-i} x^{j_1} y^{j_2} z^{j_3} (\bar{f} + ew)^{j_4} N_0^{i-j_4} \\
=& N_0^{i-j_4} x^{j_1} y^{j_2} z^{j_3} \sum_{j=0}^{j_4} \binom{j_4}{j} e^j w^{m-i+j} \bar{f}^{j_4-j} \\
=& N_0^{i-j_4} x^{j_1} y^{j_2} z^{j_3} w^{m-i} \bar{f}^{j_4} + \underbrace{N_0^{i-j_4} x^{j_1} y^{j_2} z^{j_3} \sum_{j=1}^{j_4} \binom{j_4}{j} e^j w^{m-i+j} \bar{f}^{j_4-j}}.
\end{aligned}
$$

Look at the degrees of w in the last summation, it is $m - i + j$, which is in the interval $[m - i + 1, m]$. The monomials in the summation part are in the polynomials in $\bigcup_{j=1}^{i} w^{m-i+j} G_i$. As stated before, we present the monomials in the basis according to the powers of w in it. Therefore, the monomials in the summation part already appear in the basis. The new monomials introduced are only those in the term $N_0^{i-j_4} x^{j_1} y^{j_2} z^{j_3} w^{m-i} \bar{f}^{j_4}$. The corresponding polynomials for these new monomials are in $w^{m-i} G_i$. Algorithm 1 guarantees that the final basis is triangular.

Calculating the Bound. A list of polynomials $G = \{ g_{i_1,i_2,i_3,i_4,i_5} \}$ is defined above. Denote the basis with the coefficient vectors of $g_{i_1,i_2,i_3,i_4,i_5}(wW, xX, yY, zZ)$ as its basis vectors as M and the lattice generated by M as L. Let $M^{(v,u)}$ be the submatrix whose rows are corresponding to the polynomials for some $v \in [0, m]$, $u \in [0, m - v]$ and columns are corresponding to the monomials in the form of $w^v x^u y^a z^b$ for some integers a and b. We show the structure of M in Table 1 when $m = 2$. All the entries above the main diagonal are zeroes and the entries marked as asterisks are those whose values do not contribute to the determinant.

Table 1. Structure of matrix M with $m = 2$

		w^2	w^1		w^0		
		x^0	x^0	x^1	x^0	x^1	x^2
$v = 2$	$u = 0$	$M^{(2,0)}$					
$v = 1$	$u = 0$	$*$	$M^{(1,0)}$				
	$u = 1$	$*$	$*$	$M^{(1,1)}$			
$v = 0$	$u = 0$	$*$	$*$	$*$	$M^{(0,0)}$		
	$u = 1$	$*$	$*$	$*$	$*$	$M^{(0,1)}$	
	$u = 2$	$*$	$*$	$*$	$*$	$*$	$M^{(0,2)}$

Let $s = \tau m$ for $\tau > 0$ which will be optimized later. In Appendix A, an asymptotic bound concerning the upper bounds for the sizes of the roots, W, X, Y and Z is given:

$$W^{(r+1)(1+4\tau)} X^{2(r+1)(1+2\tau)} Y^{1+4\tau+6\tau^2} Z^{r(1+4\tau+6\tau^2)} < N_0^{(r+1)(1+4\tau)}.$$

Substituting the values for N_0, W, X, Y and Z, we obtain the inequality on τ:

$$\frac{6}{r+1}\tau^2 + 4\left(\delta - \frac{1}{r+1}\right)\tau + \left(\alpha + \beta + \delta - \frac{3}{r+1}\right) < 0.$$

Let τ be the optimal value $\frac{1}{3}(1 - \delta(r+1))$. Then we obtain the inequality on δ:

$$2\delta^2 - \frac{7}{r+1}\delta + \frac{11}{(r+1)^2} - \frac{3(\alpha+\beta)}{r+1} > 0.$$

This implies that

$$\delta < \frac{7}{4(r+1)} - \frac{1}{4}\sqrt{\frac{24(\alpha+\beta)}{r+1} - \frac{39}{(r+1)^2}}.$$

The dimension of our lattice is $O(m^4)$ which is polynomial in $\frac{1}{\epsilon}$. The bit-sizes of the entries are clearly polynomial in $\log(N)$. Hence, the running time of our method is polynomial in $(\log(N), \frac{1}{\epsilon})$. The result for knowing the MSBs in Theorem 1 is obtained.

3.2 Attack with Known LSBs

In this section, we consider the case when we know some of the least significant bits of the private exponent d.

Following the notations in Section 3.1, we let $e = N^{\alpha}$ and $d = N^{\beta}$. Assume d is in the form of $d = d_1 R + \hat{d}$, where $\hat{d}$ denotes the known LSBs and R is some known integer. Let R be $N^{\beta-\delta}$. We deduce that $|d_1| = |\frac{d-\hat{d}}{R}| < |\frac{d}{R}| = N^{\delta}$. Then we can rewrite (1) as

$$e(d_1 R + \hat{d}) = 1 + k(p-1)(q-1).$$

Define

$$f_{\text{lsb}}(x, y, z) = x(y-1)(z-1) + (1 - e\hat{d}).$$

Then $(x_0, y_0, z_0) = (k, p, q)$ is a root of $f_{\text{lsb}}(x, y, z) \equiv 0 \pmod{N_1}$ where $N_1 = eR\ (= N^{\alpha+\beta-\delta})$. Define

$$X = 2N^{\alpha+\beta-2/(r+1)} \text{ and } Y = Z = 2N^{1/(r+1)},$$

then $|x_0| < X$, $|y_0| < Y$ and $|z_0| < Z$.

$f_{\text{lsb}}(x, y, z)$ contains the same monomials with the polynomial $\bar{f}$ (See Section 3.1.) considered in [12]. We can construct the lattice in an analogous way with the authors did in [12]. We can also view this problem as a special case of the problem we considered in Section 3.1. We collect polynomials as in Algorithm 2 except that we fix $v = 0$. We then construct a lattice using these polynomials. All the computations are similar to those in Appendix A except that we fix $v = 0$. We leave the calculations for the following condition in Appendix B:

$$X^{(r+1)(2+3\tau)} Y^{1+3\tau+3\tau^2} Z^{r(1+3\tau+3\tau^2)} < N_1^{(r+1)(1+3\tau)}.$$

τ is the same as in previous section. Substituting the values for N_1, X, Y and Z into the condition, we obtain

$$\delta < \frac{5}{3(r+1)} - \frac{2}{3}\sqrt{\frac{3(\alpha+\beta)}{r+1} - \frac{5}{(r+1)^2}},$$

when $\tau = \frac{1}{2}(1 - \delta(r+1))$. The running time of our method is polynomial in $(\log(N), \frac{1}{\epsilon})$ as in the previous section. This completes the proof of Theorem 2.

3.3 Attack with Known Bits in the Middle

When the known bits are in the middle of d, we can write d as $d = d_{2,1} + \bar{d}R_1 + d_{2,2}R_2$, where $\bar{d}$ represents the known bits lying in the middle of d, and $d_{2,1}$, $d_{2,2}$ represents the unknown least significant bits and most significant bits respectively. Moreover, R_1 and R_2 are two known integers. Let us assume that $d_{2,1}$ and $d_{2,2}$ are bounded by N^{δ_1} and N^{δ_2} respectively, then R_2 is about $N^{\beta-\delta_2}$. From (1) we have

$$e(d_{2,1} + \bar{d}R_1 + d_{2,2}R_2) = 1 + k(p-1)(q-1).$$

Rearranging it, we get

$$k(p-1)(q-1) - ed_{2,1} - e\bar{d}R_1 + 1 = eR_2d_{2,2}.$$

Therefore we formulate our problem as finding a small root of the polynomial

$$f_{\mathrm{mid}}(w,x,y,z) = x(y-1)(z-1) - ew + (1 - e\bar{d}R_1)$$

modulo $N_2 = eR_2$ ($\approx N^{\alpha+\beta-\delta_2}$). The root is $(w_0, x_0, y_0, z_0) = (d_{2,1}, k, p, q)$ with $|w_0| < W$, $|x_0| < X$, $|y_0| < Y$ and $|z_0| < Z$ where

$$W = N^{\delta_1},\ X = 2N^{\alpha+\beta-2/(r+1)} \text{ and } Y = Z = 2N^{1/(r+1)}.$$

The polynomial $f_{\mathrm{mid}}(w,x,y,z) = x(y-1)(z-1) - ew + (1 - e\bar{d}R_1)$ has the same monomials with f_{msb} we considered in Section 3.1. We can construct our lattice in an analogous manner as in Section 3.1 and apply the bound directly. Plugging the values for N_2, W, X, Y and Z into the bound

$$W^{(r+1)(1+4\tau)}X^{2(r+1)(1+2\tau)}Y^{1+4\tau+6\tau^2}Z^{r(1+4\tau+6\tau^2)} < N_2^{(r+1)(1+4\tau)}$$

and doing some routine calculations, we obtain that

$$\delta < \frac{7}{4(r+1)} - \frac{1}{4}\sqrt{\frac{24(\alpha+\beta)}{r+1} - \frac{39}{(r+1)^2}},$$

when $\tau = \frac{1}{3}(1 - \delta(r+1))$. Here we denote $\delta_1 + \delta_2$ as δ. The running time of our method is polynomial in $(\log(N), \frac{1}{\epsilon})$ as in Section 3.1. This completes the proof of Theorem 1.

Table 2. Some results of the experiments with $r = 2$ and $\alpha = 2/3$

	N (bits)	β	δ	m	s	$\dim(L)$	$\log_2(\det(L))$	time (LLL)
MSBs	600	0.10	0.03	6	1	280	7.69×10^5	16.2 hr
	1000	0.05	0.04	6	1	280	1.19×10^6	2.8 hr
LSBs	2000	0.20	0.05	8	2	171	2.28×10^6	54.7 hr
	1000	0.15	0.10	9	2	205	1.32×10^6	33.3 hr
MBs	1000	0.10	0.03	6	1	280	1.26×10^6	36.6 hr
	1000	0.08	0.04	6	1	280	1.21×10^6	14.0 hr

4 Experiments

Our methods are heuristic due to Assumption 1 as stated before. In order to show the correctness of our methods, we ran several experiments on a desktop running Ubuntu with 2.83GHz Intel Core2 CPU and 4GB RAM. As examples, we only ran our experiments with full-size e. Thinking of that the dimensions of our lattices are large even with small parameters (r, m, s), we chose our parameters which are relatively small.

We chose the parameters with the exact expressions, like (6) and (7) in Appendix A, for the dimensions and determinants of lattices. For a specific value of β (representing size of d), we chose a value of δ (representing size of the unknown bits) in the range of our results. Then we chose m and s subject to the condition

$$\det(L) < N_i^{m(\dim(L)-1)}$$

such that the dimension of the lattice is relatively small. We list some parameter settings and the results of the experiments in Table 2. In all our experiments, we could obtain the final roots and thus factored the moduli N.

We found that Assumption 1 may fail on a few occasions. In these cases, two ways may be used to find the final roots.

1. For example, assume we have three polynomials $f_1(x, y, z)$, $f_2(x, y, z)$ and $f_3(x, y, z)$ with a common root (x_0, y_0, z_0). We take $f_{12}(y, z) = \mathrm{Res}_x(f_1, f_2)$, $f_{13}(y, z) = \mathrm{Res}_x(f_1, f_3)$ and then $f_{23}(z) = \mathrm{Res}_y(f_{12}, f_{13})$. If unfortunately $f_{23}(z) \equiv 0$, which means that f_{12} and f_{13} have a nontrivial factor, then we can first take $f'_{12} = \frac{f_{12}}{\gcd(f_{12}, f_{13})}$ and $f'_{13} = \frac{f_{13}}{\gcd(f_{12}, f_{13})}$. Finally, we take $f'_{23}(z) = \mathrm{Res}_y(f'_{12}, f'_{13})$. It ends up with $f'_{23}(z_0) = 0$ but $f'_{23}(z) \not\equiv 0$. Use any standard root-finding algorithm to recover z_0 and then recover y_0 from $f'_{12}(y, z_0) = 0$ and x_0 from $f_1(x, y_0, z_0) = 0$.
2. Another way is to use the technique of Gröbner basis. We found that for sufficiently large N, there were more polynomials which are corresponding to the LLL-reduced basis vectors that share the desired root. This may benefit us when computing the Gröbner basis by adding all these polynomials in the basis.

Unfortunately, both the resultant computations and the Gröbner basis computations consume too much memory and time in our experiments. For some experiments, we just checked that the polynomials we obtained contain the roots indeed but rather than really did the computations.

5 Conclusion

In this paper, we considered partial key exposure attacks on Takagi's variant of RSA with moduli $N = p^r q$ $(r \geq 2)$. We presented three attacks when different parts of the private exponent are exposed to an attacker. Our results showed that when a certain number of bits of the private exponent are exposed, then the modulus N can be factored in polynomial time. We examined the validity of our methods through some experiments.

Acknowledgements. The authors would like to thank anonymous reviewers for their helpful comments and suggestions. The work of this paper was supported by the National Key Basic Research Program of China (2013CB834203), the National Natural Science Foundation of China (Grant 61070172), and the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA06010702.

References

1. Aono, Y.: Minkowski sum based lattice construction for multivariate simultaneous Coppersmith's technique and applications to RSA. In: Boyd, C., Simpson, L. (eds.) ACISP. LNCS, vol. 7959, pp. 88–103. Springer, Heidelberg (2013)
2. Blömer, J., May, A.: New partial key exposure attacks on RSA. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 27–43. Springer, Heidelberg (2003)
3. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)
4. Boneh, D., Durfee, G., Frankel, Y.: An attack on RSA given a small fraction of the private key bits. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 25–34. Springer, Heidelberg (1998)
5. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg (1996)
6. Coron, J.S., May, A.: Deterministic polynomial-time equivalence of computing the RSA secret key and factoring. J. Cryptol. 20(1), 39–50 (2007)
7. Durfee, G., Nguyên, P.Q.: Cryptanalysis of the RSA schemes with short secret exponent from Asiacrypt '99. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 14–29. Springer, Heidelberg (2000)
8. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial key exposure attacks on RSA up to full size exponents. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 371–386. Springer, Heidelberg (2005)
9. Herrmann, M., May, A.: Maximizing small root bounds by linearization and applications to small secret exponent RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 53–69. Springer, Heidelberg (2010)
10. Hinek, M.J.: Cryptanalysis of RSA and Its Variants, 1st edn. Chapman & Hall/CRC (2009)
11. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M.J. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg (1997)
12. Itoh, K., Kunihiro, N., Kurosawa, K.: Small secret key attack on a variant of RSA (due to Takagi). In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 387–406. Springer, Heidelberg (2008)
13. Jochemsz, E., May, A.: A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 267–282. Springer, Heidelberg (2006)
14. Jochemsz, E., May, A.: A polynomial time attack on RSA with private CRT-exponents smaller than $N^{0.073}$. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 395–411. Springer, Heidelberg (2007)

15. Kunihiro, N., Kurosawa, K.: Deterministic polynomial time equivalence between factoring and key-recovery attack on Takagi's RSA. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 412–425. Springer, Heidelberg (2007)
16. Lenstra, A., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)
17. May, A.: New RSA vulnerabilities using lattice reduction methods. Ph.D. thesis, University of Paderborn (2003)
18. Takagi, T.: Fast RSA-type cryptosystem modulo $p^k q$. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 318–326. Springer, Heidelberg (1998)

A The Asymptotic Bound in Section 3.1

Let $\omega_{v,u}$ be the dimension of $M^{(v,u)}$ as in Table 1. It is easy to see that

$$\omega_{v,u} = u(r+1) + (s+1) + sr = (r+1)(u+s) + 1.$$

$M^{(v,u)}$ is lower triangular and the elements on its diagonal are bounds (multiplied by some powers of N_0) for the monomials. Therefore, the determinant of $M^{(v,u)}$ is

$$\det\left(M^{(v,u)}\right) = N_0^{t_n} W^{t_w} X^{t_x} Y^{t_y} Z^{t_z},$$

where t_n, t_w, t_x, t_y and t_z are given as follows:

$$\begin{aligned}
t_n &= \sum_{i=0}^{u-1}(m-i)(r+1) + (m-u)(1+s+rs) \\
&= m((r+1)(u+s)+1) - \frac{1}{2}u((r+1)(u+2s) - r + 1), \\
t_w &= v\omega_{v,u} = v((r+1)(u+s)+1), \\
t_x &= u\omega_{v,u} = u((r+1)(u+s)+1), \\
t_y &= \sum_{i=0}^{u-1}\left(i+1+\frac{1}{2}r(r-1)\right) + \sum_{j=1}^{s}(u+j) + \sum_{k=1}^{s}\frac{1}{2}r(r-1) \\
&= \frac{1}{2}(u+s)(u+s+r(r-1)+1), \\
t_z &= \sum_{i=0}^{u-1}(i+(i+1)(r-1)) + u + \sum_{k=1}^{s}(u+k)r = \frac{1}{2}r(u+s)(u+s+1).
\end{aligned}$$

Then we can compute the dimension of the lattice L:

$$\dim(L) = \sum_{v=0}^{m}\sum_{u=0}^{m-v}\omega_{v,u} = \frac{1}{6}(r+1)(1+3\tau)m^3 + o(m^3), \tag{6}$$

and the determinant of L:

$$\det(L) = \prod_{v=0}^{m}\prod_{u=0}^{m-v}\det\left(M^{(v,u)}\right) = N_0^{s_n} W^{s_w} X^{s_x} Y^{s_y} Z^{s_z}, \tag{7}$$

where

$$s_n = \sum_{v=0}^{m}\sum_{u=0}^{m-v} t_n = \frac{1}{24}(r+1)(3+8\tau)m^4 + o(m^4),$$
$$s_w = \sum_{v=0}^{m}\sum_{u=0}^{m-v} t_w = \frac{1}{24}(r+1)(1+4\tau)m^4 + o(m^4),$$
$$s_x = \sum_{v=0}^{m}\sum_{u=0}^{m-v} t_x = \frac{1}{12}(r+1)(1+2\tau)m^4 + o(m^4),$$
$$s_y = \sum_{v=0}^{m}\sum_{u=0}^{m-v} t_y = \frac{1}{24}(1+4\tau+6\tau^2)m^4 + o(m^4),$$
$$s_z = \sum_{v=0}^{m}\sum_{u=0}^{m-v} t_z = \frac{1}{24}r(1+4\tau+6\tau^2)m^4 + o(m^4).$$

We then apply LLL-reduction algorithm to the lattice L. In order to obtain the root (w_0, x_0, y_0, z_0) by computing the resultants, we need four polynomials which all have (w_0, x_0, y_0, z_0) as a root. Since we already have two such polynomials, which are $f_1 = y^r z - N$ and $f_2 = x(y-1)(z-1) + ew + 1 - e\tilde{d}$, we need another two such polynomials. If the polynomials corresponding to the shortest two vectors in the LLL-reduced basis satisfy Howgrave-Graham's condition

$$\det(L) < N_0^{m(\dim(L)-1)},$$

we get another two such polynomials f_3 and f_4 according to Lemma 1. Then the root (w_0, x_0, y_0, z_0) can be obtained from these four polynomials by using the resultant technique under Assumption 1.

Ignore the terms that do not depend on N_0 and the low order terms $o(m^4)$, we obtain that

$$W^{(r+1)(1+4\tau)} X^{2(r+1)(1+2\tau)} Y^{1+4\tau+6\tau^2} Z^{r(1+4\tau+6\tau^2)} < N_0^{(r+1)(1+4\tau)}.$$

B The Asymptotic Bound in Section 3.2

We reuse some notations in Appendix A. The dimension of the lattice we construct for the problem in Section 3.2 is

$$\dim(L) = \sum_{u=0}^{m} \omega_{v,u} = \frac{1}{2}(r+1)(1+2\tau)m^2 + o(m^2).$$

The determinant is $\det(L) = N_1^{s_n} X^{s_x} Y^{s_y} Z^{s_z}$ where

$$s_n = \sum_{u=0}^{m} t_n = \frac{1}{6}(r+1)(2+3\tau)m^3 + o(m^3),$$

$$s_x = \sum_{u=0}^{m} t_x = \frac{1}{6}(r+1)(2+3\tau)m^3 + o(m^3),$$
$$s_y = \sum_{u=0}^{m} t_y = \frac{1}{6}(1+3\tau+3\tau^2)m^3 + o(m^3),$$
$$s_z = \sum_{u=0}^{m} t_z = \frac{1}{6}r(1+3\tau+3\tau^2)m^3 + o(m^3).$$

From $\det(L) < N_1^{m(\dim(L)-1)}$, we derive that

$$X^{(r+1)(2+3\tau)}Y^{1+3\tau+3\tau^2}Z^{r(1+3\tau+3\tau^2)} < N_1^{(r+1)(1+3\tau)}.$$