Journal of Information Security and Applications 77 (2023) 103562

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Generalized implicit-key attacks on RSA

Mengce Zheng [*]

College of Information and Intelligence Engineering, Zhejiang Wanli University, Ningbo, China
School of Information Science and Technology, University of Science and Technology of China, Hefei, China

ARTICLE INFO

MSC:
94A60

Keywords:
RSA
Cryptanalysis
Lattice
Implicit-key attack
Correlated private keys

ABSTRACT

RSA (Rivest–Shamir–Adleman) is a fundamental algorithm in information security for public key cryptography. Recently, a novel attack scenario of RSA with two implicitly correlated private keys, i.e., implicit-key attack was formulated. The lattice-based cryptanalytic strategy was proposed to factor RSA moduli using given implicit hints referring to known quantities of unknown common bits distributed among unknown private keys. In this paper, we review the simple basic scenario in which two RSA instances share known amounts of MSBs (most significant bits) and LSBs (least significant bits). We extend it to a more complex situation, where the amounts of MSBs and LSBs shared along with a few common blocks of middle bits are known. In addition, based on the above theoretical analyses, we present a generalized implicit-key attack framework. Our results disclose the vulnerability of RSA using correlated private keys with implicit information. Furthermore, numerical computer experiments are conducted to assess the validity of basic and extended implicit-key attacks.

1. Introduction

Since its invention, RSA [1] has been the most well-known cryptosystem in public key cryptography. The critical equation is $ed \equiv 1 \bmod \varphi(N)$ for the following definitions of e, d, N and $\varphi(N)$. Modulus $N = pq$ results from multiplying two prime numbers with the same bit-size. Two keys e, d are called the public and private exponents, respectively. Besides, $\varphi(N)$ is Euler's totient function and is equal to $(p-1)(q-1)$. The encryption algorithm reckons $c = m^e \bmod N$ for a plaintext m while the decryption algorithm computes $c^d \bmod N$ for a ciphertext c. Its security has been studied in [2,3] etc. and among them, Coppersmith [4] introduced the lattice-based method. After that, further variant attacks were proposed such as [5–15]. In the lattice-based method, a lattice reduction algorithm, namely the LLL algorithm [16] is always used as the main tool and small roots (associated with secret values) of modular or integer polynomial equations are the crucial targets to be solved.

The partial key exposure attack, which exposes some of the private key to the attacker, is one of the aforementioned attacks on RSA. The first analysis was done by Boneh et al. in [17]. Ernst et al. [8] proposed specific attacks that are effective up to full-size exponents according to a common heuristic assumption. Recently, more related works [12–14] have been presented. This attack type resembles the challenge of breaking RSA using an oracle that explicitly discloses several consecutive bits of d. On the other hand, the implicit factoring attack was proposed by May and Ritzenhofen [18] using an oracle that provides implicit information about p. Given two different RSA moduli $N_1 = p_1q_1$ and $N_2 = p_2q_2$ with α-bit q_i and p_1, p_2 sharing at least t many LSBs, one can recover q_1 and q_2 by the lattice-based method if $t > 2(\alpha+2)$. The attack bound was improved to $t \geq \frac{k}{k-1}\alpha$ using multiple oracle queries in the case of k RSA moduli. Following that, new results have been presented, including shared MSBs and shared middle bits [19] and other improved methods [11,15,20].

Recently, Zheng and Hu [21] concentrated on an interesting and restrictive attack scenario in which implicit knowledge about the private keys is known. This work was motivated by the implicit factoring problem and the partial key exposure attack. Although expected, side-channel attacks such as [22,23] might not provide explicit information like some disclosed private key fragments. Instead, we might readily determine a lot of implicit details about each pair of two correlated private keys. In this paper, we informally state the implicit-key attack problem as follows.

Let (N_1, e_1, d_1) and (N_2, e_2, d_2) be two different RSA instances for N_1, N_2 of the same bit-size. Assume that d_1 and d_2 are two distinct private keys with the same bit-size since MSBs of the shorter key can be padded with zero to make it true. Suppose that d_1 and d_2 have some implicit information known, namely the MSBs, LSBs and other middle bits quantities they share. Based on the knowledge of such implicitly correlated private keys, our goal is to factor N_1 and N_2 in polynomial time.

* Correspondence to: College of Information and Intelligence Engineering, Zhejiang Wanli University, Ningbo, China.
E-mail address: mczheng@zwu.edu.cn.

https://doi.org/10.1016/j.jisa.2023.103562

Available online 28 July 2023
2214-2126/© 2023 Elsevier Ltd. All rights reserved.

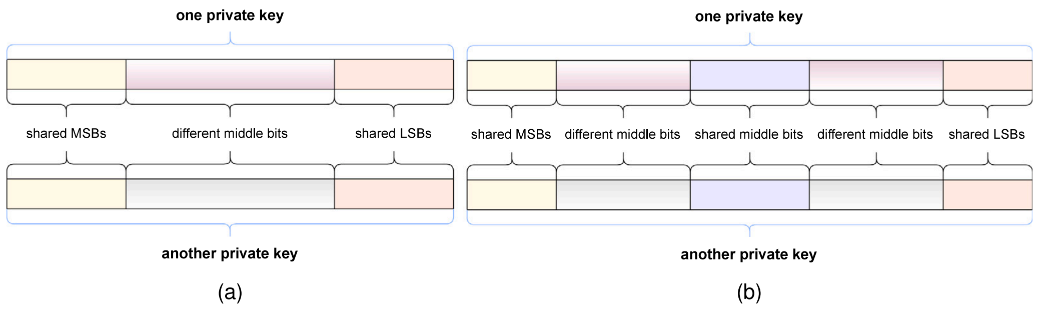

Fig. 1. Two distinct examples of two correlated private keys with implicit information: (a) basic case; (b) complex case.

The illustrative examples are depicted in Figs. 1(a) and 1(b), which divide the correlated private keys with implicit information into the basic and complex cases. More concretely, the basic case analyzed in [21], i.e., Fig. 1(a) demonstrates how d_1, d_2 share certain MSBs and LSBs, leaving one diverse block in the middle. The quantities of shared MSBs and LSBs constitute the pertinent implicit information. Comparatively, the complex case, i.e., Fig. 1(b) shows that some MSBs, LSBs, and middle bits of d_1, d_2 are shared, leaving multiple distinct middle blocks. The relevant implicit information is the amounts of several common middle bits.

The implicit-key attack is different from cryptanalyses dealing with one RSA instance. The security was investigated in [24,25] when given more RSA key pairs with the same modulus by lattice-based techniques. In our opinion, it only takes partial advantage of the implicit information about the private exponents. On the other hand, Hinek [26] analyzed another scenario in which many RSA instances using the same private key are given. The implicit information for this case is that all the private exponents are identical. Our work covers the above two special cases and makes further improvements. There are various scenarios in which two separate RSA instances are used, e.g., the application Dual RSA [27] to blind signatures and authentication. (Unfortunately, our strategy cannot be applied to Dual RSA since there is no implicit information of d_1, d_2 and even e_1, e_2 are identical.)

When two RSA instances are produced with backdoor keys [28] or imperfect randomness [29,30], we might encounter the implicit-key attack problem. Based on the theoretical interests, we consider the following topics. One is to expose the RSA vulnerability further by using weaker criteria like implicit disclosure about private keys. Moreover, we investigate how previous attacks in the literature can be extended by combining the partial key exposure attack and the implicit factorization problem.

Recently, Zheng et al. [31] studied a similar problem, which focused on the attack scenario when given two or more RSA instances having an implicit relation of the related private keys. The authors proposed lattice-based attacks by solving modular polynomial equations and applying subtle lattice techniques. We want to point out that our proposed strategy is based on solving integer polynomial equations, which is different from Zheng et al.'s attack. Besides, the prerequisites about known implicit information required in ours and [31] are different. We study the impact of various implicit relations between two correlated private keys on the security of RSA. Conversely, Zheng et al. studied the impact of the number of correlated private keys sharing a fixed implicit relation.

In our solution to the implicit-key attack problem, an integer polynomial equation is derived from two given RSA instances. The unknown variables include the sums of unknown primes and the unknown differences between two private keys. We adapt the Jochemsz–May strategy [9,32] that summarizes Coppersmith's techniques [4] and Coron's reformulation [7,10] for extracting the common root of multivariate integer polynomial equations. We finally obtain the sums of unknown primes and hence factorize given RSA moduli. To achieve theoretical effects, it relies on the following heuristic assumption. Similar to previous works like [8,9,12–15,20,31,33] in the literature, Assumption 1 holds in the practical experiments and will not be involved in the propositions below.

Assumption 1. Our lattice-based attacks yield algebraically independent integer polynomial equations, and the common root can be efficiently solved using the Gröbner basis computation.

Unless otherwise specified, $N = 2^l$ throughout this work refers to an integer of the same bit-size as the given RSA moduli. The proposed attack result in [21] with respect to the basic implicit-key attack on standard RSA is expressed in Proposition 1. Because the relevant lattice dimension may preferably be large, it should be noted that the offered theoretical findings are asymptotic. Fortunately, the experimental results using lattices with low dimensions as provided in [21] are very close to the theoretical ones.

Proposition 1 *([21]). Let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two distinct RSA moduli of the same bit-size l, where primes p_1, q_1, p_2, q_2 are of the same bit-size $l/2$. Let e_1, e_2, d_1, d_2 satisfy $e_1d_1 \equiv 1 \bmod \varphi(N_1)$ and $e_2d_2 \equiv 1 \bmod \varphi(N_2)$, such that e_1, e_2 and d_1, d_2 have the same bit-size l and δl, respectively. Suppose that d_1 and d_2 share $\beta_1 l$ MSBs and $\beta_2 l$ LSBs. Then N_1 and N_2 can be factored in polynomial time if*

$$\delta < \frac{(\beta+1)(1+10\tau+20\tau^2)-10\tau^2-30\tau^3}{4+30\tau+40\tau^2}, \tag{1}$$

where $\beta = \beta_1 + \beta_2$ and $\tau \geq 0$. Let τ_0 denote the unique positive root that satisfies

$$120x^4 + 180x^3 + (46-20\beta)x^2 - 8\beta x - \beta - 1 = 0. \tag{2}$$

Hence, the above condition on δ reaches its maximal upper bound that is

$$\delta < \frac{(\beta+1)(1+10\tau_0+20\tau_0^2)-10\tau_0^2-30\tau_0^3}{4+30\tau_0+40\tau_0^2}.$$

The parameter τ_0 is the optimal value that leads to the maximum of the right side of (1). This value can be calculated using numerical methods. Our insecure bound on δ for $\beta = 0$ is $\delta < 0.280$ (using the optimized $\tau_0 = 0.120$), which is weaker than Boneh–Durfee bound $\delta < 0.292$ reported in [5]. It happens because we make use of two RSA instances with no implicit information instead of only one instance. Since more unknown variables might weaken the upper bound on δ, Boneh–Durfee attack works more efficiently when little or even no implicit information is given. However, our approach has the advantage of being more flexible in terms of attack cases and is useful in certain situations where the Boneh–Durfee attack is not applicable.

We develop the findings and strategy in the previous work [21]. In addition to the basic attack, we have analyzed two special cases and further propose an extended attack. In particular, we extend the implicit-key attack to a more challenging scenario in which we have any number n of unknown middle blocks. However, as n increases, our strategy becomes less effective. One explanation is that the running time of such an attack is exponential in parameter n. Another reason is that when dealing with more middle blocks, the upper bounds on the unknown variables are smaller. In order to verify the validity of the extended implicit-key attack, we provide the asymptotic cryptanalytic results by numerical experiments for $n = 2$.

Our contribution mainly includes the generalized implicit-key attacks on RSA and the validation experimental results, which are summarized as follows. We study the security of RSA with two implicitly correlated private keys and propose the implicit-key attack problem. We propose generalized implicit-key attacks of factoring RSA modulus concerning its theoretical security issue. We verify the correctness and validity of the proposed implicit-key attacks with numerical computer experiments and always successfully obtain the factorization results.

The rest is organized as follows. Section 2 outlines the requirements for obtaining the desired common root by using the lattice reduction algorithms. The basic implicit-key attack when considering the basic case described in Fig. 1(a) is reviewed in Section 3. The extended implicit-key attack when handling the complex case described in Fig. 1(b) is presented in Section 4. Section 5 provides validation results with detailed comparison according to extensive numerical experiments. Finally, Section 6 concludes the paper.

2. Preliminaries

We introduce the LLL algorithm [16] and the lattice-based method that was proposed by Coppersmith [4] and further improved by Jochemsz and May [9]. The condition for finding the roots of integer polynomial equations is given as a mathematical consequence of the lattice-based method. For more information, see [3,34].

The collection of all integer linear combinations of linearly independent vectors $\vec{b}_1, \dots, \vec{b}_m$ is referred to as a lattice $\mathcal{L}$. Thus, it can be written as

$$\mathcal{L}(\vec{b}_1, \dots, \vec{b}_m) = \left\{ \sum_{i=1}^{m} z_i \vec{b}_i \,:\, z_i \in \mathbb{Z} \right\}.$$

We construct an $m \times n$ basis matrix B by treating each n-dimensional basis vector $\vec{b}_i$ as a row. The lattice determinant is $\det(\mathcal{L}) = \sqrt{\det(BB^{\top})}$. We only take into account a full-rank lattice with $m = n$ since it simplifies the subsequent analysis and improves the attack efficiency. This is a common simplification in the lattice-based method and does not affect the general definition of lattice determinant. Hence, we get $\det(\mathcal{L}) = |\det(B)|$ because B is a square matrix.

Due to its effective running performance, the LLL algorithm [16] is used to find approximately short lattice vectors. The approximately short lattice vectors refer to the LLL-reduced lattice vectors used in the lattice-based method. An approximately short lattice vector is a vector whose length is close (within some constant multiples) to the shortest non-zero vector in the lattice. These vectors play a crucial role because they allow us to further derive several integer equations we are trying to solve. Given an m-dimensional lattice basis vector $\vec{v}_i = (v_{i_1}, \dots, v_{i_m})$, its length (i.e., Euclidean norm) is defined as $\|\vec{v}_i\| := (\sum_{j=1}^{m} v_{i_j}^2)^{1/2}$. Lemma 1 is used to establish the relationship between the lengths of the reduced lattice vectors and the lattice determinant. On the basis of its outputs, the following lemma (that was proven in [34, Theorem 4]) is presented.

Lemma 1 (*[34]*). *Let lattice $\mathcal{L}$ be spanned by a given basis $(\vec{b}_1, \vec{b}_2, \dots, \vec{b}_m)$. A reduced basis $(\vec{v}_1, \vec{v}_2, \dots, \vec{v}_m)$ is derived from the LLL algorithm satisfying*

$$\|\vec{v}_i\| \le 2^{\frac{m(m-1)}{4(m+1-i)}} \det(\mathcal{L})^{\frac{1}{m+1-i}}, \ 1 \le i \le m.$$

The running time is polynomial in the maximal component of input vectors and lattice dimension m.

The following lemma (i.e., [34, Theorem 14]) is a direct generalization of Howgrave-Graham's lemma [35], which provides a rule for figuring out whether a modular equation's root is also a root over the integers. Given a polynomial $g(x_1, \dots, x_n) = \sum a_{i_1,\dots,i_n} x_1^{i_1} \cdots x_n^{i_n}$, its norm is defined as $\|g(x_1, \dots, x_n)\| := (\sum |a_{i_1,\dots,i_n}|^2)^{1/2}$.

Lemma 2 (*[34]*). *Let $g(x_1, \dots, x_n) \in \mathbb{Z}[x_1, \dots, x_n]$ be an integer polynomial, which is a sum of at most m monomials. Suppose that*

1. *$g(x_1^{(0)}, \dots, x_n^{(0)}) \equiv 0 \bmod R$ for a positive integer R, where $|x_1^{(0)}| < X_1, \dots, |x_n^{(0)}| < X_n$,*
2. *$\|g(x_1X_1, \dots, x_nX_n)\| < \frac{R}{\sqrt{m}}$.*

Then we have $g(x_1^{(0)}, \dots, x_n^{(0)}) = 0$ over the integers.

Thus, to solve modular or integer polynomial equations using Lemmas 1 and 2, we make use of the first ℓ many vectors outputted by the LLL algorithm. It follows that one can construct an integer equation system to solve the unknown variables if

$$2^{\frac{m(m-1)}{4(m+1-\ell)}} \det(\mathcal{L})^{\frac{1}{m+1-\ell}} < \frac{R}{\sqrt{m}}.$$

It leads to

$$\det(\mathcal{L}) < R^{m+1-\ell} 2^{-\frac{m(m-1)}{4}} m^{-\frac{m+1-\ell}{2}}.$$

As $\ell < m \ll R$, with a small error term ϵ, it can be further simplified to $\det(\mathcal{L}) \le R^{m-\epsilon}$. Hence, we roughly derive a simplified asymptotic condition

$$\det(\mathcal{L}) < R^m. \tag{3}$$

The Jochemsz–May strategy [9] is used to construct a triangular lattice basis matrix. The lattice determinant $\det(\mathcal{L})$ is calculated as the product of the diagonal elements of the constructed basis matrix. We provide a general condition for solving small roots of integer polynomial equations and sketch the lattice-based method. In practice, several unknowns in RSA instances lead to an integer polynomial equation used in the proposed lattice-based attacks. Because the lattice-based method always regards a polynomial equation as a polynomial for convenient writing and symbolic calculation, we may use polynomial instead of polynomial equation in this sense. More concretely, we want to find the root of an h-variate integer polynomial $f(x_1, \dots, x_h) = \sum a_{i_1,\dots,i_h} x_1^{i_1} \cdots x_h^{i_h}$ when conducting the proposed implicit-key attack.

First, as mentioned in Lemma 2, we need to compute the upper bounds X_i on unknown variables x_i for $i = 1, \dots, h$. Moreover, X_∞ is defined as the highest potential value of a single term in $f(x_1, x_2, \dots, x_h)$. That is

$$\begin{aligned} X_\infty &= \|f(x_1X_1, x_2X_2, \dots, x_hX_h)\|_\infty \\ &= \max\left\{ |a_{i_1,i_2,\dots,i_h}| X_1^{i_1} X_2^{i_2} \cdots X_h^{i_h} \right\}. \end{aligned} \tag{4}$$

We then define

$$R = X_\infty (X_1 X_2 \cdots X_{h-2})^{s-1} (X_{h-1} X_h)^{s-1+t} \tag{5}$$

for two non-negative integers s and t to be determined in the subsequent lattice construction.

Then we adapt the extended Jochemsz–May strategy [9] for finding small integer roots and use extra shifts of two variables x_{h-1} and x_h. A lattice basis matrix is constructed via the coefficient vectors of the shift polynomials, which is derived from two monomial sets S and S_R. To do so, we define

$$S = \bigcup_{0 \le j_{h-1}, j_h \le t} \left\{ x_1^{i_1} x_2^{i_2} \cdots x_{h-2}^{i_{h-2}} x_{h-1}^{i_{h-1}+j_{h-1}} x_h^{i_h+j_h} \,:\, x_1^{i_1} x_2^{i_2} \cdots x_h^{i_h} \in f^{s-1} \right\},$$

and

$$S_R = \bigcup_{0 \le j_{h-1}, j_h \le t} \left\{ x_1^{i_1} x_2^{i_2} \cdots x_{h-2}^{i_{h-2}} x_{h-1}^{i_{h-1}+j_{h-1}} x_h^{i_h+j_h} \,:\, x_1^{i_1} x_2^{i_2} \cdots x_h^{i_h} \in f^{s} \right\}.$$

The parameters s and t are defined as two non-negative integers satisfying $s \ge 1$ and $t \ge 0$, which are used to control the number of elements in the monomial sets S and S_R and also control the dimension of the constructed lattice.

The constructions of specific shift polynomials will be given in Section 3 and Section 4, respectively. Based on our lattice construction, condition (3) finally reduces to $\prod_{i=1}^{h} X_i^{s_i} < X_\infty^{s_g}$. Here $s_i =$

$\sum_{x_1^{k_1}\cdots x_h^{k_h}\in S_R\setminus S} k_i$ is the sum of exponent k_i over variable x_i and $s_g = |S|$ is the cardinality of monomial set S. More details refer to the analysis in Section 3.

Similar to the usage description of the lattice-based method in the multivariate case in [34, Page 47]. We summarize the lattice-based method to solve a multivariate integer polynomial equation in four steps. The first step is to use $f(x_1,\dots,x_h)$ and known quantities for generating two monomial sets S, S_R and constructing shift polynomials $g(x_1,\dots,x_h)$ and $g'(x_1,\dots,x_h)$ having a common root $(x_1',\dots,x_h')$ modulo R. The second step is to let $\vec{b}_i$ be a row vector derived from the coefficient vector of $g(x_1X_1,\dots,x_hX_h)$ and $g'(x_1X_1,\dots,x_hX_h)$ for all $1\le i\le m$. Hence, one can generate the lattice $\mathcal{L}=\left\{\sum_{i=1}^m z_i\vec{b}_i : z_i\in\mathbb{Z}\right\}$. The third step is to apply the LLL algorithm on $\mathcal{L}$. One can obtain the first ℓ many reduced basis vectors $\vec{v}_1,\dots,\vec{v}_\ell$ and transform the vectors to integer polynomials $f_1(x_1,\dots,x_h),\dots,f_\ell(x_1,\dots,x_h)$ sharing the common root $(x_1',\dots,x_h')$ over $\mathbb{Z}$. The transformation is based on inverse corresponding relationship used in the second step for converting the coefficient vectors to lattice basis vectors. The last step is to check if the derived polynomials $f_i(x_1,\dots,x_h)$ for $1\le i\le\ell$ along with the original polynomial f are algebraically independent. If so, the equation system $f_i(x_1,\dots,x_h)=0$ (including $f(x_1,\dots,x_h)=0$) can be solved using the Gröbner basis computation. Hence, one extracts the desired root $(x_1',\dots,x_h')$.

Under the above process, the first ℓ many reduced vectors are obtained. We discover a collection of polynomials $f_1,\dots,f_\ell$ having the shared root over the integers. Then the Gröbner basis computation is employed for extracting the common root since it is efficient for more variables. The running time mainly depends on computing the reduced lattice basis matrix and recovering the desired root. For the basic implicit-key attack, both of them can be done in polynomial time. For the extended implicit-key attack, the running time is exponential in the number of unknown middle blocks n.

3. Basic implicit-key attack

We review the basic implicit-key attack for two RSA instances (N_1,e_1,d_1) and (N_2,e_2,d_2). We first consider the general case when e_1,e_2 are of arbitrary bit-size and d_1,d_2 share some MSBs and LSBs leaving one different block in the middle. We start by considering the typical scenario in which d_1,d_2 share some MSBs and LSBs, leaving one diverse block in the middle, and e_1,e_2 are of arbitrary bit-size. Later we focus on two special cases.

3.1. The general case

We show the following result for the general case in the basic implicit-key attack, which was already presented in [21, Theorem 1].

Proposition 2 (*[21]*). *Let $N_1=p_1q_1$ and $N_2=p_2q_2$ be two distinct RSA moduli of the same bit-size l, where primes p_1,q_1,p_2,q_2 are of the same bit-size $l/2$. Let e_1,e_2,d_1,d_2 satisfy $e_1d_1\equiv 1 \bmod \varphi(N_1)$ and $e_2d_2\equiv 1 \bmod \varphi(N_2)$, such that e_1, e_2 and d_1,d_2 have bit-size $\alpha_1 l$, $\alpha_2 l$ and δl, respectively. Suppose that d_1 and d_2 share $\beta_1 l$ MSBs and $\beta_2 l$ LSBs. Then N_1 and N_2 can be factored in polynomial time if*

$$\delta<\frac{(\alpha+\beta-1)(1+10\tau+20\tau^2)-10\tau^2-30\tau^3}{4+30\tau+40\tau^2}-\frac{\alpha}{2}+1, \tag{6}$$

where $\alpha=\alpha_1+\alpha_2$, $\beta=\beta_1+\beta_2$ and $\tau\ge 0$. Let τ_0 denote the only positive real root satisfying

$$\begin{aligned}&120x^4+180x^3+(86-20\alpha-20\beta)x^2\\&+(16-8\alpha-8\beta)x-\alpha-\beta+1=0.\end{aligned}\tag{7}$$

The above condition on δ reaches its maximal upper bound that is

$$\delta<\frac{(\alpha+\beta-1)(1+10\tau_0+20\tau_0^2)-10\tau_0^2-30\tau_0^3}{4+30\tau_0+40\tau_0^2}-\frac{\alpha}{2}+1.$$

Proof. The key equation $ed\equiv 1 \bmod \varphi(N)$, with two positive integers k_1 and k_2, gives us

$$\begin{cases}e_1d_1=k_1(N_1+1-p_1-q_1)+1,\\ e_2d_2=k_2(N_2+1-p_2-q_2)+1.\end{cases}$$

Multiplying two equations by e_2,e_1 respectively and then subtracting, we have

$$\begin{aligned}&e_1e_2(d_1-d_2)\\&=e_2k_1(N_1+1-p_1-q_1)+e_2-e_1k_2(N_2+1-p_2-q_2)-e_1.\end{aligned}\tag{8}$$

Note that N_1,N_2 are of the same bit-size l for $N=2^l$ as mentioned above. Consider that we are given d_1 and d_2 of the same bit-size δl sharing $\beta_1 l$ MSBs and $\beta_2 l$ LSBs. Additionally, β_1 or β_2 can be estimated as 0. Hence, it implies that

$$\begin{cases}d_1=d_{00}2^{(\delta-\beta_1)l}+d_{11}2^{\beta_2 l}+d_{01},\\ d_2=d_{00}2^{(\delta-\beta_1)l}+d_{21}2^{\beta_2 l}+d_{01}.\end{cases}\tag{9}$$

Here d_{00} and d_{01} denote the respective shared MSBs and LSBs whilst d_{11} and d_{21} denote two different middle bit-blocks of the private keys. Substituting (9) into (8), it can be rewritten as

$$\begin{aligned}&e_1e_2(d_{11}-d_{21})2^{\beta_2 l}\\&=e_2k_1(N_1+1-p_1-q_1)-e_1k_2(N_2+1-p_2-q_2)+e_2-e_1.\end{aligned}$$

It further reduces to

$$\begin{aligned}&e_1e_22^{\beta_2 l}(d_{21}-d_{11})+e_2(N_1+1)k_1-e_1(N_2+1)k_2\\&-e_2k_1(p_1+q_1)+e_1k_2(p_2+q_2)+(e_2-e_1)=0.\end{aligned}$$

We list the known variables (denoted by a_i) and the unknown ones (denoted by x_i) as follows.

$$\begin{cases}a_1=e_1e_22^{\beta_2 l},\\ a_2=e_2(N_1+1),\\ a_3=-e_1(N_2+1),\\ a_4=-e_2,\\ a_5=e_1,\\ a_6=e_2-e_1.\end{cases}\quad\text{and}\quad\begin{cases}x_1=d_{21}-d_{11},\\ x_2=k_1,\\ x_3=k_2,\\ x_4=p_1+q_1,\\ x_5=p_2+q_2.\end{cases}$$

Our goal is to discover the solution to the following integer polynomial,

$$\begin{aligned}&f(x_1,x_2,x_3,x_4,x_5)\\&=a_1x_1+a_2x_2+a_3x_3+a_4x_2x_4+a_5x_3x_5+a_6.\end{aligned}\tag{10}$$

We simply do division to make the polynomial irreducible if e_1 and e_2 have a nontrivial common divisor. In order to apply Coppersmith's techniques, the norm of the unknown variables should be small enough.

The upper bounds X_i on each x_i are calculated as follows for given $e_1=N^{\alpha_1}$ and $e_2=N^{\alpha_2}$ of arbitrary bit-size. Let $\beta=\beta_1+\beta_2$, we have

$$X_1=N^{\delta-\beta},\ X_2=N^{\alpha_1+\delta-1},\ X_3=N^{\alpha_2+\delta-1},\ X_4=X_5=N^{1/2}.\tag{11}$$

As we introduced the definition of the maximal norm X_∞ in (4), it can be computed as

$$X_\infty=N^{\alpha+\delta},\ \alpha=\alpha_1+\alpha_2.\tag{12}$$

We use two extra shifts of x_4 and x_5 for solving the above integer polynomial (10). For two non-negative integers $s\ge 1$ and $t\ge 0$, the following monomial sets S and S_R are built.

$$S=\bigcup_{0\le j_4,j_5\le t}\left\{x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4+j_4}x_5^{i_5+j_5} : x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5}\in f^{s-1}\right\},$$

$$S_R=\bigcup_{0\le j_4,j_5\le t}\left\{x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4+j_4}x_5^{i_5+j_5} : x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5}\in f^{s}\right\}.$$

By computing the expansion of f^{s-1} and f^s, we know the relationship between monomials $x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5}$ in S, S_R and the corresponding exponents i_1, i_2, i_3, i_4, i_5. We have

$$x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5} \in S \Leftrightarrow \begin{cases} i_1 = 0,\dots,s-1, \\ i_2 = 0,\dots,s-1-i_1, \\ i_3 = 0,\dots,s-1-i_1-i_2, \\ i_4 = 0,\dots,i_2+t, \\ i_5 = 0,\dots,i_3+t. \end{cases}$$

$$x_1^{i_1}x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5} \in S_R \Leftrightarrow \begin{cases} i_1 = 0,\dots,s, \\ i_2 = 0,\dots,s-i_1, \\ i_3 = 0,\dots,s-i_1-i_2, \\ i_4 = 0,\dots,i_2+t, \\ i_5 = 0,\dots,i_3+t. \end{cases}$$

We require the constant term of $f(x_1,x_2,x_3,x_4,x_5)$, i.e., a_6 to be 1. Thus, we define a modular polynomial $f' = a_6^{-1} f \bmod R$, where

$$R = X_\infty X_1^{s-1} X_2^{s-1} X_3^{s-1} X_4^{s-1+t} X_5^{s-1+t} \tag{13}$$

as mentioned in (5). The shift polynomials g and g' according to S and S_R are defined as follows.

$$g : \frac{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} f' \cdot R}{X_\infty X_1^{k_1}X_2^{k_2}X_3^{k_3}X_4^{k_4}X_5^{k_5}}, \quad x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S,$$
$$g' : x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} R, \quad x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S.$$

The coefficient vectors of g and g' are used in the construction of lattice $\mathcal{L}$, where x_iX_i is substituted for each x_i. As discussed in Section 2, we need to compute $\det(\mathcal{L})$ to apply the condition (3). In our lattice construction, the diagonal elements of g and g' are equal to $\frac{R}{X_\infty}$ and $X_1^{k_1}X_2^{k_2}X_3^{k_3}X_4^{k_4}X_5^{k_5}R$, respectively. So it implies

$$\left(\frac{R}{X_\infty}\right)^{s_g} X_1^{s_1}X_2^{s_2}X_3^{s_3}X_4^{s_4}X_5^{s_5} R^{s_R} < R^m,$$

where $s_g = |S|$, $s_i = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_i$, $s_R = |S_R \setminus S|$ and $m = |S_R|$. Furthermore, we have $m = |S_R| = |S| + |S_R \setminus S| = s_g + s_R$. Hence, it can be finally reduced to

$$X_1^{s_1}X_2^{s_2}X_3^{s_3}X_4^{s_4}X_5^{s_5} < X_\infty^{s_g}, \tag{14}$$

where $s_i = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_i$ and $s_g = |S|$.

We now calculate s_i for $i = 1,2,3,4,5$ and s_g, m by the above deduction. Based on their definitions, we have

$$s_1 = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_1$$
$$= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_1 - \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_1$$
$$= \frac{s^5}{120} + \frac{s^4t}{12} + \frac{s^3t^2}{6} + o(s^5),$$
$$s_2 = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_2$$
$$= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_2 - \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_2$$
$$= \frac{s^5}{60} + \frac{s^4t}{8} + \frac{s^3t^2}{6} + o(s^5),$$
$$s_3 = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_3$$
$$= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_3 - \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_3$$
$$= \frac{s^5}{60} + \frac{s^4t}{8} + \frac{s^3t^2}{6} + o(s^5),$$
$$s_4 = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_4$$
$$= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_4 - \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_4$$
$$= \frac{s^5}{120} + \frac{s^4t}{12} + \frac{s^3t^2}{4} + \frac{s^2t^3}{4} + o(s^5),$$
$$s_5 = \sum_{x_1^{k_1}x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_5$$
$$= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_5 - \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_5$$
$$= \frac{s^5}{120} + \frac{s^4t}{12} + \frac{s^3t^2}{4} + \frac{s^2t^3}{4} + o(s^5),$$
$$s_g = |S| = \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\sum_{i_3=0}^{s-1-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} 1$$
$$= \frac{s^5}{120} + \frac{s^4t}{12} + \frac{s^3t^2}{6} + o(s^5),$$
$$m = |S_R| = \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\sum_{i_3=0}^{s-i_1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} 1$$
$$= \frac{s^5}{120} + \frac{s^4t}{12} + \frac{s^3t^2}{6} + o(s^5).$$

After tedious computation and taking $t = \tau s$ for $\tau \geq 0$ and omitting lower terms $o(s^5)$, we obtain

$$\begin{aligned} s_1 = s_g &= \frac{1}{120}(1+10\tau+20\tau^2)s^5, \\ s_2 = s_3 &= \frac{1}{120}(2+15\tau+20\tau^2)s^5, \\ s_4 = s_5 &= \frac{1}{120}(1+10\tau+30\tau^2+30\tau^3)s^5 \end{aligned} \tag{15}$$

The values of X_i, X_∞, s_i and s_g are substituted into the condition (14) and it gives

$$N^{\frac{1}{120}(\delta-\beta)(1+10\tau+20\tau^2)s^5} N^{\frac{1}{120}(\alpha_1+\delta-1)(2+15\tau+20\tau^2)s^5}$$
$$\times N^{\frac{1}{120}(\alpha_2+\delta-1)(2+15\tau+20\tau^2)s^5} N^{\frac{1}{240}(1+10\tau+30\tau^2+30\tau^3)s^5}$$
$$\times N^{\frac{1}{240}(1+10\tau+30\tau^2+30\tau^3)s^5} < N^{\frac{1}{120}(\alpha+\delta)(1+10\tau+20\tau^2)s^5}.$$

We deal with the exponents over N (eliminating the term s^5) and obtain

$$(\delta-\beta)(1+10\tau+20\tau^2) + (\alpha+2\delta-2)(2+15\tau+20\tau^2)$$
$$+1+10\tau+30\tau^2+30\tau^3 < (\alpha+\delta)(1+10\tau+20\tau^2).$$

This results in

$$\delta < \frac{(\alpha+\beta-1)(1+10\tau+20\tau^2) - 10\tau^2 - 30\tau^3}{4+30\tau+40\tau^2} - \frac{\alpha}{2} + 1.$$

For known α and β, there exists an optimal τ_0 maximizing the right side. By calculating the derivative with respect to τ, we take the unique positive τ_0 satisfying

$$120x^4 + 180x^3 + (86-20\alpha-20\beta)x^2 + (16-8\alpha-8\beta)x - \alpha - \beta + 1 = 0.$$

Hence, the above condition on δ reaches its maximal upper bound that is

$$\delta < \frac{(\alpha+\beta-1)(1+10\tau_0+20\tau_0^2) - 10\tau_0^2 - 30\tau_0^3}{4+30\tau_0+40\tau_0^2} - \frac{\alpha}{2} + 1.$$

We follow the four-step summary of the lattice-based method and finally derive four polynomials f_1, f_2, f_3, f_4 separate from f under the above analysis and condition. Additionally, f, f_1, f_2, f_3 and f_4 share the common root $(d_{21}-d_{11}, k_1, k_2, p_1+q_1, p_2+q_2)$ over the integers. Finally, we extract $p_1 + q_1$ and $p_2 + q_2$, which result in the factorization of N_1 and N_2 respectively.

The running time is mainly dominated by the LLL algorithm, which is polynomial in the maximal component of input vectors and lattice dimension as stated in Lemma 1. The maximal component of input vectors related to multiples of X_i and X_∞ is polynomial in N and the lattice dimension m is polynomial in s^5. Thus, the time complexity is polynomial in N, s and the factorization works in polynomial time. □

We briefly explain how to prove Proposition 1 since it is a special case of Proposition 2. Suppose that e_1, e_2 are of full bit-size, i.e., $\alpha = \alpha_1 + \alpha_2 = 1 + 1 = 2$, N_1 and N_2 can be factored in polynomial time if

$$\delta < \frac{(\beta+1)(1+10\tau+20\tau^2) - 10\tau^2 - 30\tau^3}{4+30\tau+40\tau^2},$$

where $\beta = \beta_1 + \beta_2$ and $\tau \ge 0$. If the unique positive root τ_0 satisfies

$$120x^4 + 180x^3 + (46 - 20\beta)x^2 - 8\beta x - \beta - 1 = 0,$$

we obtain the maximal upper bound on δ that is

$$\delta < \frac{(\beta+1)(1+10\tau_0+20\tau_0^2) - 10\tau_0^2 - 30\tau_0^3}{4+30\tau_0+40\tau_0^2}.$$

3.2. Two special cases

We focus on two special cases, namely given two RSA instances with a common modulus or a common private key. The implicit-key attack for two RSA instances with a given common modulus is presented first.

Proposition 3. *Let $N = pq$ be an RSA modulus of bit-size l, where primes p, q are of the same bit-size $l/2$. Let e_1, e_2, d_1, d_2 satisfy $e_1 d_1 \equiv 1 \bmod \varphi(N)$ and $e_2 d_2 \equiv 1 \bmod \varphi(N)$, such that e_1, e_2 and d_1, d_2 are of bit-size $\alpha_1 l$, $\alpha_2 l$ and δl, respectively. Suppose that d_1 and d_2 share $\beta_1 l$ MSBs and $\beta_2 l$ LSBs. Then N can be factored in polynomial time if*

$$\delta < \frac{(\alpha+\beta)(4+8\tau) - 3 - 8\tau - 6\tau^2}{4(3+4\tau)} - \frac{\alpha}{2} + 1,$$

where $\alpha = \alpha_1 + \alpha_2$, $\beta = \beta_1 + \beta_2$ and $\tau \ge 0$. Let

$$\tau_0 = \frac{-9 + \sqrt{48(\alpha+\beta)+9}}{12}$$

that maximizes the right side of the above condition on δ. By substituting τ_0 into the above condition, it reaches the maximal upper bound that is

$$\delta < \frac{8\beta + 17 - \sqrt{48(\alpha+\beta)+9}}{16}. \tag{16}$$

Proof. Based on the analysis of the general case, we have a new integer polynomial for the same modulus N,

$$f_N(x_1, x_2, x_3, x_4) = a_1 x_1 + a_2 x_2 + a_3 x_3 + a_4 x_2 x_4 + a_5 x_3 x_4 + a_6.$$

The known and unknown variables are identical to those introduced in Section 3.1 except for $x_4 = x_5$ and $N_1 = N_2 = N$. Besides, the upper bounds X_i and X_∞ are the same as (11) and (12). Concretely, we have $X_1 = N^{\delta-\beta}$, $X_2 = N^{\alpha_1+\delta-1}$, $X_3 = N^{\alpha_2+\delta-1}$, $X_4 = N^{1/2}$ and $X_\infty = N^{\alpha+\delta}$.

We then build a refined lattice to discover the root of $f_N(x_1, x_2, x_3, x_4)$. The procedure is similar and we skip over its detailed construction. We show the monomials that belong to S and S_R for two negative integers $s \ge 1$ and $t \ge 0$ as follows.

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

$$x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \in S_R \Leftrightarrow \begin{cases} i_1 = 0, \dots, s, \\ i_2 = 0, \dots, s-i_1, \\ i_3 = 0, \dots, s-i_1-i_2, \\ i_4 = 0, \dots, i_2+i_3+t. \end{cases}$$

Similarly, we calculate each s_i for $i = 1, 2, 3, 4$ and s_g, m based on their definitions as follows.

$$\begin{aligned}
s_1 &= \sum_{x_1^{k_1} x_2^{k_2} x_3^{k_3} x_4^{k_4} \in S_R \setminus S} k_1 \\
&= \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} \sum_{i_3=0}^{s-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_1 - \sum_{i_1=0}^{s-1} \sum_{i_2=0}^{s-1-i_1} \sum_{i_3=0}^{s-1-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_1 \\
&= \frac{s^4}{12} + \frac{s^3 t}{6} + o(s^4), \\
s_2 &= \sum_{x_1^{k_1} x_2^{k_2} x_3^{k_3} x_4^{k_4} \in S_R \setminus S} k_2 \\
&= \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} \sum_{i_3=0}^{s-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_2 - \sum_{i_1=0}^{s-1} \sum_{i_2=0}^{s-1-i_1} \sum_{i_3=0}^{s-1-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_2 \\
&= \frac{s^4}{8} + \frac{s^3 t}{6} + o(s^4), \\
s_3 &= \sum_{x_1^{k_1} x_2^{k_2} x_3^{k_3} x_4^{k_4} \in S_R \setminus S} k_3 \\
&= \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} \sum_{i_3=0}^{s-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_3 - \sum_{i_1=0}^{s-1} \sum_{i_2=0}^{s-1-i_1} \sum_{i_3=0}^{s-1-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_3 \\
&= \frac{s^4}{8} + \frac{s^3 t}{6} + o(s^4), \\
s_4 &= \sum_{x_1^{k_1} x_2^{k_2} x_3^{k_3} x_4^{k_4} \in S_R \setminus S} k_4 \\
&= \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} \sum_{i_3=0}^{s-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_4 - \sum_{i_1=0}^{s-1} \sum_{i_2=0}^{s-1-i_1} \sum_{i_3=0}^{s-1-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} i_4 \\
&= \frac{s^4}{8} + \frac{s^3 t}{3} + \frac{s^2 t^2}{4} + o(s^4), \\
s_g &= |S| = \sum_{i_1=0}^{s-1} \sum_{i_2=0}^{s-1-i_1} \sum_{i_3=0}^{s-1-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} 1 \\
&= \frac{s^4}{12} + \frac{s^3 t}{6} + o(s^4), \\
m &= |S_R| = \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} \sum_{i_3=0}^{s-i_1-i_2} \sum_{i_4=0}^{i_2+i_3+t} 1 \\
&= \frac{s^4}{12} + \frac{s^3 t}{6} + o(s^4).
\end{aligned}$$

By taking $t = \tau s$ for $\tau \ge 0$, we have

$$\begin{aligned}
s_1 &= s_g = \frac{1}{12}(1+2\tau)s^4, \\
s_2 &= s_3 = \frac{1}{24}(3+4\tau)s^4, \\
s_4 &= \frac{1}{24}(3+8\tau+6\tau^2)s^4.
\end{aligned}$$

Substituting X_i, s_i, X_∞ and s_g into $X_1^{s_1} X_2^{s_2} X_3^{s_3} X_4^{s_4} < X_\infty^{s_g}$ and dealing with the exponents over N, we obtain

$$(\delta - \beta)(2+4\tau) + (\alpha + 2\delta - 2)(3+4\tau) + \frac{3+8\tau+6\tau^2}{2} < (\alpha+\delta)(2+4\tau).$$

This results in

$$\delta < \frac{(\alpha+\beta)(4+8\tau) - 3 - 8\tau - 6\tau^2}{4(3+4\tau)} - \frac{\alpha}{2} + 1.$$

By calculating the derivative concerning τ, the right side can be maximized at

$$\tau_0 = \frac{-9 + \sqrt{48(\alpha+\beta)+9}}{12}$$

and plugging this optimized τ_0 in the above inequality for δ gives

$$\delta < \frac{8\beta + 17 - \sqrt{48(\alpha+\beta)+9}}{16}.$$

We follow the four-step summary of the lattice-based method and finally obtain three polynomials f_1, f_2, f_3 apart from f_N under the above analysis and condition. The polynomials share the common root $(d_{21} - d_{11}, k_1, k_2, p+q)$ over the integers. Finally, $p+q$ is extracted and directly leads to the factorization of N.

The running time is mainly dominated by the LLL algorithm, which is polynomial in the maximal component of input vectors and lattice dimension as stated in Lemma 1. The maximal component of input vectors related to multiples of X_i and X_∞ is polynomial in N and the lattice dimension m is polynomial in s^4. Thus, the time complexity is polynomial in N, s and the factorization works in polynomial time. □

The implicit-key attack for two RSA instances with a shared private key is then presented.

Proposition 4. *Let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two distinct RSA moduli of the same bit-size l, where primes p_1, q_1, p_2, q_2 are of the same bit-size $l/2$. Let e_1, e_2, d satisfy $e_1 d \equiv 1 \bmod \varphi(N_1)$ and $e_2 d \equiv 1 \bmod \varphi(N_2)$, such that e_1, e_2 and d are of bit-size $\alpha_1 l$, $\alpha_2 l$ and δl, respectively. Then N_1 and N_2 can be factored in polynomial time if*

$$\delta < \frac{3 - \alpha + (16-4\alpha)\tau + 6\tau^2 - 12\tau^3}{3 + 16\tau + 12\tau^2}, \tag{17}$$

where $\alpha = \alpha_1 + \alpha_2$ and $\tau \ge 0$ Let τ_0 denote the unique positive τ that satisfies

$$36x^4 + 96x^3 + (51 - 12\alpha)x^2 + (9 - 6\alpha)x - \alpha = 0. \tag{18}$$

Hence, the above condition on δ reaches its maximal upper bound that is

$$\delta < \frac{3 - \alpha + (16-4\alpha)\tau_0 + 6\tau_0^2 - 12\tau_0^3}{3 + 16\tau_0 + 12\tau_0^2}.$$

Proof. Based on the analysis of the general case, we have a new integer polynomial for the same private key d,

$$f_d(x_2, x_3, x_4, x_5) = a_2x_2 + a_3x_3 + a_4x_2x_4 + a_5x_3x_5 + a_6.$$

The known and unknown variables are identical to those introduced in Section 3.1 except for $x_1 = 0$. Besides, the upper bounds X_i and X_∞ are the same as (11) and (12). Thus, we have $X_2 = N^{\alpha_1+\delta-1}$, $X_3 = N^{\alpha_2+\delta-1}$, $X_4 = X_5 = N^{1/2}$, and $X_\infty = N^{\alpha+\delta}$.

We then build another refined lattice to find the root of $f_d(x_2, x_3, x_4, x_5)$. We show the monomials that belong to S and S_R for two negative integers $s \ge 1$ and $t \ge 0$ as follows and skip over its detailed construction as well.

$$x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5} \in S \Leftrightarrow \begin{cases} i_2 = 0, \ldots, s-1, \\ i_3 = 0, \ldots, s-1-i_2, \\ i_4 = 0, \ldots, i_2 + t, \\ i_5 = 0, \ldots, i_3 + t. \end{cases}$$

$$x_2^{i_2}x_3^{i_3}x_4^{i_4}x_5^{i_5} \in S_R \Leftrightarrow \begin{cases} i_2 = 0, \ldots, s, \\ i_3 = 0, \ldots, s-i_2, \\ i_4 = 0, \ldots, i_2 + t, \\ i_5 = 0, \ldots, i_3 + t. \end{cases}$$

Similarly, we calculate each s_i for $i = 2, 3, 4, 5$ and s_g, m based on their definitions as follows.

$$\begin{aligned} s_2 &= \sum_{x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_2 \\ &= \sum_{i_2=0}^{s}\sum_{i_3=0}^{s-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_2 - \sum_{i_2=0}^{s-1}\sum_{i_3=0}^{s-1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_2 \\ &= \frac{s^4}{12} + \frac{s^3t}{2} + \frac{s^2t^2}{2} + o(s^4), \\ s_3 &= \sum_{x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_3 \\ &= \sum_{i_2=0}^{s}\sum_{i_3=0}^{s-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_3 - \sum_{i_2=0}^{s-1}\sum_{i_3=0}^{s-1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_3 \\ &= \frac{s^4}{12} + \frac{s^3t}{2} + \frac{s^2t^2}{2} + o(s^4), \\ s_4 &= \sum_{x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_4 \\ &= \sum_{i_2=0}^{s}\sum_{i_3=0}^{s-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_4 - \sum_{i_2=0}^{s-1}\sum_{i_3=0}^{s-1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_4 \\ &= \frac{s^4}{24} + \frac{s^3t}{3} + \frac{3s^2t^2}{4} + \frac{st^3}{2} + o(s^4), \\ s_5 &= \sum_{x_2^{k_2}x_3^{k_3}x_4^{k_4}x_5^{k_5} \in S_R \setminus S} k_5 \\ &= \sum_{i_2=0}^{s}\sum_{i_3=0}^{s-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_5 - \sum_{i_2=0}^{s-1}\sum_{i_3=0}^{s-1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} i_5 \\ &= \frac{s^4}{24} + \frac{s^3t}{3} + \frac{3s^2t^2}{4} + \frac{st^3}{2} + o(s^4), \\ s_g &= |S| = \sum_{i_2=0}^{s-1}\sum_{i_3=0}^{s-1-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} 1 \\ &= \frac{s^4}{24} + \frac{s^3t}{3} + \frac{s^2t^2}{2} + o(s^4), \\ m = |S_R| &= \sum_{i_2=0}^{s}\sum_{i_3=0}^{s-i_2}\sum_{i_4=0}^{i_2+t}\sum_{i_5=0}^{i_3+t} 1 \\ &= \frac{s^4}{24} + \frac{s^3t}{3} + \frac{s^2t^2}{2} + o(s^4). \end{aligned}$$

By taking $t = \tau s$ for $\tau \ge 0$, we have

$$\begin{aligned} s_g &= \frac{1}{24}(1 + 8\tau + 12\tau^2)s^4, \\ s_2 = s_3 &= \frac{1}{12}(1 + 6\tau + 6\tau^2)s^4, \\ s_4 = s_5 &= \frac{1}{24}(1 + 8\tau + 18\tau^2 + 12\tau^3)s^4. \end{aligned}$$

Substituting X_i, s_i, X_∞ and s_g into $X_2^{s_2}X_3^{s_3}X_4^{s_4}X_5^{s_5} < X_\infty^{s_g}$ and dealing with the exponents over N, we obtain

$$2(\alpha + 2\delta - 2)(1 + 6\tau + 6\tau^2) + 1 + 8\tau + 18\tau^2 + 12\tau^3 < (\alpha + \delta)(1 + 8\tau + 12\tau^2).$$

It leads to

$$\delta < \frac{3 - \alpha + (16-4\alpha)\tau + 6\tau^2 - 12\tau^3}{3 + 16\tau + 12\tau^2}.$$

For known α, there exists an optimal τ_0 maximizing the right side. By calculating the derivative with respect to τ, τ_0 is the unique positive root satisfying

$$36x^4 + 96x^3 + (51 - 12\alpha)x^2 + (9 - 6\alpha)x - \alpha = 0.$$

Hence, the above condition on δ reaches its maximal upper bound that is

$$\delta < \frac{3 - \alpha + (16-4\alpha)\tau_0 + 6\tau_0^2 - 12\tau_0^3}{3 + 16\tau_0 + 12\tau_0^2}.$$

We follow the four-step summary of the lattice-based method and finally obtain three polynomials f_1, f_2, f_3 apart from f_d under the above analysis and condition. The polynomials share the common root

Table 1
The comparison of our results and previous ones on the insecure bound of δ.

Common modulus N		Common private key d	
Previous result [24, Theorem 1][a]	$\delta < \frac{1}{3}\beta_1 + \frac{5}{12}$	Previous result [26, Formula (4.3)][b]	$\delta < \frac{1}{3} - \log_N 6$
Ours ($\beta = \beta_1 + \beta_2$)	$\delta < \frac{8\beta+17-\sqrt{48\beta+105}}{16}$	Ours ($\tau_0 \approx 0.229$)	$\delta < 0.411$

[a]The original bound is $\frac{1}{12}\beta^* + \frac{1}{6}\delta - \frac{5}{48} < 0$ for $d_1,\ d_2 < N^{\delta}$ and $|d_1 - d_2| < N^{\beta^*}$. For clear comparison, we should replace β^* with $\delta - \beta_1$ since $|d_1 - d_2| < N^{\delta-\beta_1}$ in our interpretation and hence we obtain $\delta < \frac{1}{3}\beta_1 + \frac{5}{12}$.
[b]The original bound is $\delta < \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_r} 6$, where r is the number of given RSA instances and N_r is the rth RSA modulus. For clear comparison, we should replace r with 2 since two RSA instances are considered in our attack and N is used to denote N_r as explained above. Hence, we obtain $\delta < \frac{1}{3} - \log_N 6$.

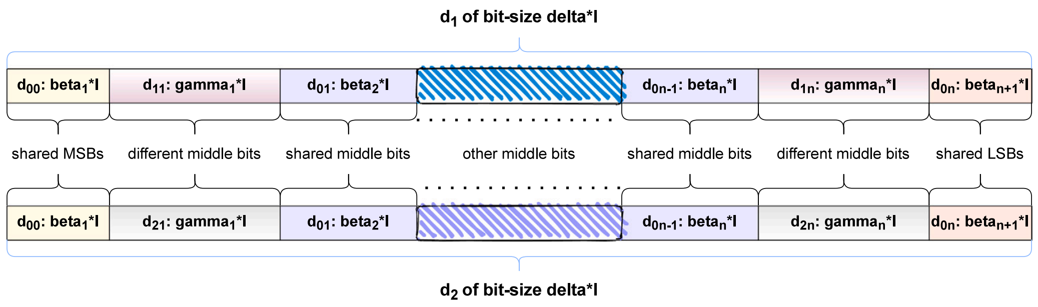

Fig. 2. The illustration of two correlated private keys with implicit information in the complex case.

$(k_1, k_2, p_1+q_1, p_2+q_2)$ over the integers. Finally, we extract p_1+q_1 and p_2+q_2, which result in the factorization of N_1 and N_2, respectively.

The running time is mainly dominated by the LLL algorithm, which is polynomial in the maximal component of input vectors and lattice dimension as stated in Lemma 1. The maximal component of input vectors related to multiples of X_i and X_∞ is polynomial in N and the lattice dimension m is polynomial in s^4. Thus, the time complexity is polynomial in N, s and the factorization works in polynomial time. □

We compare our theoretical results with previous ones for full bit-size public exponents, i.e., $\alpha_1 = \alpha_2 = 1$ and hence $\alpha = 2$. The comparison is showed in Table 1. It is clear that our results of two special cases are superior since we use more implicit information along with an improved approach.

4. Extended implicit-key attack

We generalize the solving strategy for the basic case when analyzing one block in the middle to the complex case when analyzing n discrete middle blocks. Suppose that the private keys d_1 and d_2 are of the same bit-size δl. Moreover, they share $\beta_1 l$ MSBs, $\beta_{n+1} l$ LSBs and other middle blocks of $\beta_2 l, \ldots, \beta_n l$ bits leaving n different middle blocks of $\gamma_1 l, \gamma_2 l, \ldots, \gamma_n l$ bits behind. The detailed illustration is showed in Fig. 2. To be specific, we have

$$\begin{cases} d_1 = d_{00}2^{(\delta-\beta_1)l} + d_{11}2^{(\delta-(\beta_1+\gamma_1))l} + \cdots + d_{0n}, \\ d_2 = d_{00}2^{(\delta-\beta_1)l} + d_{21}2^{(\delta-(\beta_1+\gamma_1))l} + \cdots + d_{0n}. \end{cases} \tag{19}$$

Moreover, we have $d_2 - d_1 = \sum_{i=1}^{n}(d_{2i} - d_{1i})2^{\eta_i l}$, where $\eta_i = \delta - \sum_{j=1}^{i}(\beta_j + \gamma_j)$ and d_{1i}, d_{2i} denote the ith matching different bit blocks. We substitute $d_2 - d_1$ into the RSA Eq. (8) involving two RSA instances. Hence, our aim is to find the root $(d_{21} - d_{11}, \ldots, d_{2n} - d_{1n}, k_1, k_2, p_1 + q_1, p_2 + q_2)$ of the integer polynomial

$$f(x_1, x_2, \ldots, x_{n+4}) = \sum_{i=1}^{n+2} a_i x_i + \sum_{i=n+3}^{n+4} a_i x_{i-2} x_i + a_{n+5}. \tag{20}$$

Similar to the analysis in Section 3.1, we list all the unknown and known variables below.

$$\begin{cases} a_1 = e_1 e_2 2^{\eta_1 l}, \\ \quad\vdots \\ a_n = e_1 e_2 2^{\eta_n l}, \\ a_{n+1} = e_2(N_1+1), \\ a_{n+2} = -e_1(N_2+1), \\ a_{n+3} = -e_2, \\ a_{n+4} = e_1, \\ a_{n+5} = e_2 - e_1. \end{cases} \quad \text{and} \quad \begin{cases} x_1 = d_{21} - d_{11}, \\ \quad\vdots \\ x_n = d_{2n} - d_{1n}, \\ x_{n+1} = k_1, \\ x_{n+2} = k_2, \\ x_{n+3} = p_1 + q_1, \\ x_{n+4} = p_2 + q_2. \end{cases}$$

We primarily consider arbitrary public exponents $e_1 = N^{\alpha_1}$ and $e_2 = N^{\alpha_2}$. The upper bounds X_i and X_∞ are fixed as follows.

$$\begin{aligned} &X_1 = N^{\gamma_1},\ X_2 = N^{\gamma_2},\ \cdots,\ X_n = N^{\gamma_n}, \\ &X_{n+1} = N^{\alpha_1+\delta-1},\ X_{n+2} = N^{\alpha_2+\delta-1}, \\ &X_{n+3} = X_{n+4} = N^{1/2},\ X_\infty = N^{\alpha+\delta},\ \alpha = \alpha_1 + \alpha_2. \end{aligned} \tag{21}$$

We adapt the extended Jochemsz–May strategy and define the following monomial sets

$$S = \bigcup_{0 \le j_{n+3}, j_{n+4} \le t} \{x_1^{i_1} \cdots x_{n+3}^{i_{n+3}+j_{n+3}} x_{n+4}^{i_{n+4}+j_{n+4}} : x_1^{i_1} \cdots x_{n+4}^{i_{n+4}} \in f^{s-1}\},$$

and

$$S_R = \bigcup_{0 \le j_{n+3}, j_{n+4} \le t} \{x_1^{i_1} \cdots x_{n+3}^{i_{n+3}+j_{n+3}} x_{n+4}^{i_{n+4}+j_{n+4}} : x_1^{i_1} \cdots x_{n+4}^{i_{n+4}} \in f^{s}\}.$$

The parameters s and t are two non-negative integers satisfying $s \ge 1$ and $t \ge 0$, which are used to control the number of elements in the monomial sets S and S_R and also control the dimension of the constructed lattice.

By generalizing the relation between a monomial and its indices, we know that the monomial element $x_1^{i_1}x_2^{i_2}\cdots x_{n+4}^{i_{n+4}}\in S$ relates to

$$\begin{cases} i_1 = 0,\dots,s-1,\\ i_2 = 0,\dots,s-1-i_1,\\ \quad\vdots\\ i_{n+1} = 0,\dots,s-1-i_1-\cdots-i_n,\\ i_{n+2} = 0,\dots,s-1-i_1-\cdots-i_{n+1},\\ i_{n+3} = 0,\dots,i_{n+1}+t,\\ i_{n+4} = 0,\dots,i_{n+2}+t.\end{cases}$$

and $x_1^{i_1}x_2^{i_2}\cdots x_{n+4}^{i_{n+4}}\in S_R$ relates to

$$\begin{cases} i_1 = 0,\dots,s,\\ i_2 = 0,\dots,s-i_1,\\ \quad\vdots\\ i_{n+1} = 0,\dots,s-i_1-\cdots-i_n,\\ i_{n+2} = 0,\dots,s-i_1-\cdots-i_{n+1},\\ i_{n+3} = 0,\dots,i_{n+1}+t,\\ i_{n+4} = 0,\dots,i_{n+2}+t.\end{cases}$$

The constructions of $g_{k_1,k_2,\dots,k_{n+4}}$ and $g'_{k_1,k_2,\dots,k_{n+4}}$ are straightforward and similar to that in Section 3.1. We provide the condition for finding $(n+3)$ many polynomials $f_1,f_2,\dots,f_{n+3}$ sharing the common root over the integers. That is

$$\prod_{i=1}^{n+4} X_i^{s_i} < X_\infty^{s_g}, \tag{22}$$

where $s_i=\sum_{x_1^{k_1}\cdots x_{n+4}^{k_{n+4}}\in S_R\setminus S} k_i$ and $s_g=|S|$.

Note that $s_1,s_2,\dots,s_n$ are equal based on the structure of the original polynomial (20). Moreover, we also know $s_{n+1}=s_{n+2}$ and $s_{n+3}=s_{n+4}$. Similar to the index calculation used in Appendix of [25], we conduct tedious computation of $s_1,s_2,\dots,s_{n+4}$ and s_g,m based on their definitions as follows.

$$\begin{aligned} s_1 &= s_2=\cdots=s_n\\ &= \sum_{x_1^{k_1}\cdots x_{n+4}^{k_{n+4}}\in S_R\setminus S} k_1\\ &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\cdots\sum_{i_{n+1}=0}^{s-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_1\\ &\quad-\sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\cdots\sum_{i_{n+1}=0}^{s-1-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-1-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_1\\ &= \frac{s^{n+4}+2(n+4)s^{n+3}t+(n+3)(n+4)s^{n+2}t^2}{(n+4)!}+o(s^{n+4}),\end{aligned}$$

$$\begin{aligned} s_{n+1} &= s_{n+2}\\ &= \sum_{x_1^{k_1}\cdots x_{n+4}^{k_{n+4}}\in S_R\setminus S} k_{n+1}\\ &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\cdots\sum_{i_{n+1}=0}^{s-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_{n+1}\\ &\quad-\sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\cdots\sum_{i_{n+1}=0}^{s-1-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-1-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_{n+1}\\ &= \frac{2s^{n+4}+3(n+4)s^{n+3}t+(n+3)(n+4)s^{n+2}t^2}{(n+4)!}+o(s^{n+4}),\end{aligned}$$

$$\begin{aligned} s_{n+3} &= s_{n+4}\\ &= \sum_{x_1^{k_1}\cdots x_{n+4}^{k_{n+4}}\in S_R\setminus S} k_{n+3}\\ &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\cdots\sum_{i_{n+1}=0}^{s-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_{n+3}\\ &\quad-\sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\cdots\sum_{i_{n+1}=0}^{s-1-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-1-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} i_{n+3}\\ &= \frac{2s^{n+4}+4(n+4)s^{n+3}t+3(n+3)(n+4)s^{n+2}t^2}{2(n+4)!}\\ &\quad+\frac{(n+2)(n+3)(n+4)s^{n+1}t^3}{2(n+4)!}+o(s^{n+4}),\end{aligned}$$

$$\begin{aligned} s_g &= |S|\\ &= \sum_{i_1=0}^{s-1}\sum_{i_2=0}^{s-1-i_1}\cdots\sum_{i_{n+1}=0}^{s-1-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-1-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} 1\\ &= \frac{s^{n+4}+2(n+4)s^{n+3}t+(n+3)(n+4)s^{n+2}t^2}{(n+4)!}+o(s^{n+4}),\end{aligned}$$

$$\begin{aligned} m &= |S_R|\\ &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}\cdots\sum_{i_{n+1}=0}^{s-i_1-\cdots-i_n}\sum_{i_{n+2}=0}^{s-i_1-\cdots-i_{n+1}}\sum_{i_{n+3}=0}^{i_{n+1}+t}\sum_{i_{n+4}=0}^{i_{n+2}+t} 1\\ &= \frac{s^{n+4}+2(n+4)s^{n+3}t+(n+3)(n+4)s^{n+2}t^2}{(n+4)!}+o(s^{n+4}).\end{aligned}$$

By taking $t=\tau s$ for $\tau\ge 0$, we obtain

$$\begin{aligned} s_1 &= s_2=\cdots=s_n=s_g\\ &= \frac{1+2(n+4)\tau+(n+3)(n+4)\tau^2}{(n+4)!}s^{n+4},\end{aligned} \tag{23}$$

$$s_{n+1}=s_{n+2}=\frac{2+3(n+4)\tau+(n+3)(n+4)\tau^2}{(n+4)!}s^{n+4}, \tag{24}$$

$$\begin{aligned} s_{n+3}=s_{n+4} &= \frac{2+4(n+4)\tau+3(n+3)(n+4)\tau^2}{2(n+4)!}s^{n+4}\\ &+\frac{(n+2)(n+3)(n+4)\tau^3}{2(n+4)!}s^{n+4}.\end{aligned} \tag{25}$$

Substituting $X_1,\dots,X_{n+4},X_\infty$ and $s_1,\dots,s_{n+4},s_g$ into the condition (22) and dealing with the exponents over N, we obtain

$$\begin{aligned} &(\delta-\beta)(1+2(n+4)\tau+(n+3)(n+4)\tau^2)\\ &+(\alpha+2\delta-2)(2+3(n+4)\tau+(n+3)(n+4)\tau^2)\\ &+1+2(n+4)\tau+\frac{3}{2}(n+3)(n+4)\tau^2+\frac{1}{2}(n+2)(n+3)(n+4)\tau^3\\ &<(\alpha+\delta)(1+2(n+4)\tau+(n+3)(n+4)\tau^2),\end{aligned}$$

where $\alpha=\alpha_1+\alpha_2$ and $\beta=\sum_{i=1}^n\beta_i$. By denoting

$$\begin{aligned} \lambda_{\alpha,\beta,n}(\tau) &= \frac{(\alpha+\beta-1)(1+2(n+4)\tau+(n+3)(n+4)\tau^2)}{2(2+3(n+4)\tau+(n+3)(n+4)\tau^2)}\\ &-\frac{(n+3)(n+4)\tau^2+(n+2)(n+3)(n+4)\tau^3}{4(2+3(n+4)\tau+(n+3)(n+4)\tau^2)}-\frac{\alpha}{2}+1,\end{aligned} \tag{26}$$

it further reduces to $\delta<\lambda_{\alpha,\beta,n}(\tau)$ for given α,β,n and $\tau\ge 0$.

Definition 1. Let $d(\tau)$ be the derivative of $\lambda_{\alpha,\beta,n}(\tau)$ with respect to τ for known α,β,n. It is defined as

$$\begin{aligned} d(\tau) &= \frac{\partial\lambda_{\alpha,\beta,n}(\tau)}{\partial\tau}=(n+2)(n+3)^2(n+4)\tau^4+6(n+2)(n+3)(n+4)\tau^3\\ &+((11-2\alpha-2\beta)(n+3)(n+4)-12(n+3))\tau^2\\ &+(8-4\alpha-4\beta)(n+3)\tau-2(\alpha+\beta-1).\end{aligned}$$

Table 2
The numerical examples with particular parameters of RSA instances.

n	α_1	α_2	α	β_1	β_2	β_3	β_4	β	τ_0	δ
1	0.75	1	1.75	0.05	0.05	–	–	0.1	0.102	< 0.360
	1	1	2	0.05	0.05	–	–	0.1	0.131	< 0.311
	1	1	2	0.1	0.15	–	–	0.25	0.148	< 0.357
	1	1	2	0.15	0.15	–	–	0.3	0.153	< 0.373
	1	1.25	2.25	0.15	0.15	–	–	0.3	0.181	< 0.328
2	0.75	1	1.75	0.02	0.02	0.02	–	0.06	0.077	< 0.347
	1	1	2	0.02	0.02	0.02	–	0.06	0.099	< 0.297
	1	1	2	0.04	0.05	0.06	–	0.15	0.107	< 0.325
	1	1	2	0.1	0.1	0.1	–	0.3	0.120	< 0.371
	1	1.25	2.25	0.1	0.1	0.1	–	0.3	0.141	< 0.326
3	0.75	1	1.75	0.01	0.02	0.02	0.03	0.08	0.065	< 0.352
	1	1	2	0.01	0.02	0.02	0.03	0.08	0.083	< 0.302
	1	1	2	0.03	0.03	0.04	0.04	0.14	0.088	< 0.321
	1	1	2	0.07	0.07	0.07	0.1	0.31	0.100	< 0.373
	1	1.25	2.25	0.07	0.07	0.07	0.1	0.31	0.117	< 0.327

In order to maximize $\lambda_{\alpha,\beta,n}(\tau)$, we denote by τ_0 the unique positive root of $d(\tau)$ for given α, β, n, namely $d(\tau_0) = 0$ with real $\tau_0 > 0$. Hence, $\lambda_{\alpha,\beta,n}(\tau)$ reaches its maximum when taking $\tau = \tau_0$, which can be calculated using numerical methods.

Similarly, we follow the four-step summary of the lattice-based method and finally obtain sufficient polynomials apart from the integer polynomial (20) under the above analysis and condition. All these polynomials share the common root $(d_{21} - d_{11}, \dots, d_{2n} - d_{1n}, k_1, k_2, p_1 + q_1, p_2 + q_2)$ over the integers. Thus, we extract $p_1 + q_1$ and $p_2 + q_2$, which lead to the factorization of N_1 and N_2, respectively.

The running time is mainly dominated by the LLL algorithm, which is polynomial in the maximal component of input vectors and lattice dimension as stated in Lemma 1. The maximal component of input vectors related to multiples of X_i and X_∞ is polynomial in N and the lattice dimension m is polynomial in s^{n+4}. Thus, the time complexity is polynomial in $N(= 2^l)$ and s^n. The factorization works in time that is polynomial in 2^l but exponential in n. The attack result is stated below.

Proposition 5. *Let $N_1 = p_1q_1$ and $N_2 = p_2q_2$ be two distinct RSA moduli of the same bit-size l, where primes p_1, q_1, p_2, q_2 are of the same bit-size $l/2$. Let e_1, e_2, d_1, d_2 satisfy $e_1d_1 \equiv 1 \bmod \varphi(N_1)$ and $e_2d_2 \equiv 1 \bmod \varphi(N_2)$, such that e_1, e_2 and d_1, d_2 are of bit-size $\alpha_1 l$, $\alpha_2 l$ and δl, respectively. Suppose that d_1 and d_2 share $\beta_1 l$ MSBs, $\beta_{n+1} l$ LSBs and other middle blocks of $\beta_2 l, \dots, \beta_n l$ bits, leaving n many middle blocks of $\gamma_1 l, \dots, \gamma_n l$ bits behind. Then N_1 and N_2 can be factored if*

$$\delta < \lambda_{\alpha,\beta,n}(\tau), \tag{27}$$

where $\alpha = \alpha_1 + \alpha_2$, $\beta = \sum_{i=1}^{n} \beta_i$ and $\tau \geq 0$. Let τ_0 be the unique positive root of $d(\tau)$. Hence, the above condition on δ reaches its maximal upper bound that is

$$\delta < \lambda_{\alpha,\beta,n}(\tau_0).$$

The running time is polynomial in 2^l but exponential in n.

Revisiting the basic case of $n = 1$, we immediately have the following bound

$$\delta < \frac{(\alpha + \beta - 1)(1 + 10\tau + 20\tau^2)}{2(2 + 15\tau + 20\tau^2)} - \frac{20\tau^2 + 60\tau^3}{4(2 + 15\tau + 20\tau^2)} - \frac{\alpha}{2} + 1.$$

This bound is identical to that presented in (6), which means our extended implicit-key attack is a natural generalization of the basic one.

Although we analyze the situation when n discrete blocks in the middle of d_1, d_2 are different, this attack may be inefficient in practice since the lattice dimension becomes much larger and the time consumption increase higher as n gets greater. We provide several numerical examples with particular parameters of RSA instances in Table 2 for intuitive display and understanding.

The implicit-key attacks might be used by hackers or cybercriminals who have obtained partial access to the RSA backdoor key generation through means such as data manipulation or social engineering. To be specific, two RSA backdoor private keys d_1 and d_2 are generated with several predetermined implicit relations under the control of hackers or cybercriminals. They can use the implicit information known about d_1 and d_2, i.e., given assumptions about the amounts of shared MSBs, LSBs or middle bits to factor the private keys and gain complete access to the encrypted communication. This could potentially compromise sensitive information, such as financial transactions or personal information, and cause significant harm to the individuals or groups involved.

5. Validation experiments

Before providing the experimental results, we give a simplified strategy for implementing the proposed implicit-key attacks. The asymptotic bounds stated in Proposition 2 are reached by $\tau = t/s < 0.2$ according to our numerical calculation. The value of s should be fixed at least 6 for $t = 1$. The resulting lattice dimension $m = 966$ seems inefficient to perform our simulated experiments. Therefore, our simplified strategy is based on taking $t = 0$ for implementation, which makes the lattice construction easier whereas the corresponding upper bound on δ is lower. In the following simulated experiments, we mainly choose $t = 0$ (i.e. $\tau = 0$) for efficient validation. The simulated numerical experiments are meant to demonstrate the practical feasibility and efficiency of the proposed attacks under the corresponding viable conditions.

We give experimental findings to demonstrate the effectiveness of the aforementioned attacks according to Proposition 2, Proposition 3, Proposition 4 and Proposition 5, respectively. The experiments were conducted under Windows 10 running on a computer with Intel Core i5-10500 CPU 3.10 GHz and 8 GB RAM. We utilized the LLL algorithm and the Gröbner basis computation available in `SageMath` [36]. The numbers used in each experiment were uniformly and randomly produced. We searched for the best experimental results, i.e., the highest δ values so that we could conduct a successful attack on generated RSA instances.

We were able to gather significantly more polynomials that met our solvable requirements during the experiments. In other words, we got more sufficiently short vectors than we wanted after executing the LLL algorithm. Therefore, using the Gröbner basis computation, we could find the common root and then factor the given RSA moduli. We would like to point out that although the practical performance of the LLL algorithm is better than expected, the theoretical asymptotic bounds of δ are slightly higher than the experimental ones. We show a toy numerical example to provide a clear understanding of our proposed attacks and the attack performance.

Example 1. In order to check the correctness and validity of generalized implicit-key attacks on RSA, i.e., Proposition 5, we choose the following specific parameters and generate the test example.

1. Randomly generate two 128-bit prime numbers p_1, q_1 and the modulus $N_1 = p_1 q_1$ (that implies $l = 256$);
2. Randomly generate two 128-bit prime numbers p_2, q_2 and the modulus $N_2 = p_2 q_2$ (that implies $l = 256$);
3. Randomly generate two 75-bit private keys d_1 and d_2 sharing 10-bit MSBs, 15-bit LSBs and one middle block of 25 bits, leaving two middle blocks of 12 and 13 bits behind;
4. Compute the public keys e_1 and e_2 based on above parameters.

The values of the numerical example are as follows.

$$\begin{aligned}
N_1 &= 77074460308761357021887758135198551463\backslash\\
&\quad 7926486674652585577153046769417463957,\\
N_2 &= 68208572317566179670508749924351550957\backslash\\
&\quad 3217051127796887822196933203004728293,\\
e_1 &= 65881643579810762787606479542022726581\backslash\\
&\quad 3538164718606821351701327168166148501,\\
e_2 &= 46442954770397161194672969713167544936\backslash\\
&\quad 0109718417539401882425858344035554889.
\end{aligned}$$

To apply generalized implicit-key attack proposed in Proposition 5, we choose $s = 3,\ t = 0$, which means that we need to apply the LLL algorithm to a lattice $\mathcal{L}$ with dimension $m = 84$. After running for about 8 seconds, the approximately shortest basis vectors that meet the solvable condition are obtained. The system of integer equations to be solved is then derived by transforming reduced vectors into integer polynomials. We finally solve it by applying the Gröbner basis computation in less than one second and recover the unknown variables as follows.

$$\begin{aligned}
x_1 &= 1220,\\
x_2 &= 360,\\
x_3 &= 2345187345418626413367 3,\\
x_4 &= 1868867027683410479753 6,\\
x_5 &= 56460479126778773018971233160417267380 2,\\
x_6 &= 52303766900340535090407468871960023999 4.
\end{aligned}$$

Thus, we know the values of $p_1 + q_1$ and $p_2 + q_2$ through x_5 and x_6.

$$\begin{aligned}
p_1 + q_1 &= 564604791267787730189712331604172673802,\\
p_2 + q_2 &= 523037669003405350904074688719600239994.
\end{aligned}$$

Eventually, we extract p_1, q_1, p_2 and q_2 based on $p_1 + q_1$, $p_2 + q_2$, N_1 and N_2.

$$\begin{aligned}
p_1 &= 231114679608716682316576127775596020963,\\
q_1 &= 333490111659071047873136203828576652839,\\
p_2 &= 247971566084683177310935343410715903843,\\
q_2 &= 275066102918722173593139345308884336151.
\end{aligned}$$

One may check that $N_1 = p_1 q_1$ and $N_2 = p_2 q_2$ do hold, so the generalized implicit-key attack proposed in Proposition 5 successfully outputs the factorization of N_1 and N_2.

5.1. Experimental results for Proposition 2

For our basic implicit-key attack, we generated two l-bit (i.e., $l = 1024,\ 2048,\ 3072$) RSA moduli and two public exponents (denoted by α_1, α_2 and $\alpha = \alpha_1 + \alpha_2$) appeared in the experiments were nearly of full bit-size. The implicit information of shared MSBs and LSBs were indicated by β_1, β_2 and $\beta = \beta_1 + \beta_2$. Because the lattice construction was fixed by $s = 3,\ t = 0$, we needed to reduce a 56-dimension lattice. Table 3 displays the comparison of experimental insecure bounds. The theoretical upper bound on δ for specific parameters in our simplified strategy is provided by the δ_t-column. The experimental upper bound on δ for specific parameters in our computer experiments is provided by the δ_e-column. The AR-column indicates the achieving rate calculated as δ_e/δ_t, which compares our experimental bound with the theoretical one. We use m to represent the matching lattice dimension, and Time to denote the running time (measured in seconds).

We gathered enough polynomials with a common root over the integers for each experiment. More polynomials were added to the Gröbner basis computation since the first four might be insufficient for extracting the common root. Finally, we were able to determine the correct values for $p_1 + q_1$ and $p_2 + q_2$, which factorized N_1 and N_2, respectively. We established the value of x_3 before determining the solution to the other variables if the Gröbner basis computation did not immediately produce the desired root. According to Table 3, $m = 56$ is sufficient since the experimental bound is so close to the theoretical one, which is based on the observation that the average achieving rate is 98.8%.

5.2. Experimental results for Proposition 3

For our particular implicit-key attack against RSA with shared modulus N and d_1, d_2 having some common MSBs and LSBs, we generated an l-bit (i.e., $l = 1024,\ 2048,\ 3072$) RSA modulus and two public exponents (denoted by α_1, α_2 and $\alpha = \alpha_1 + \alpha_2$) appeared in the experiments were nearly of full bit-size. The implicit information of shared MSBs and LSBs were indicated by β_1, β_2 and $\beta = \beta_1 + \beta_2$. Because the lattice construction was fixed by $s = 2, 3, 4$ and $t = 0$, we needed to reduce lattices whose dimensions are 20, 50 and 105, respectively. Table 4 displays the comparison of experimental insecure bounds. The theoretical upper bound on δ for specific parameters in our simplified strategy is provided by the δ_t-column. The experimental upper bound on δ for specific parameters in our computer experiments is provided by the δ_e-column. The AR-column indicates the achieving rate calculated as δ_e/δ_t, which compares our experimental bound with the theoretical one. We use m to represent the matching lattice dimension, and Time to denote the running time (measured in seconds).

We gathered enough polynomials with a common root over the integers for each experiment. More polynomials were added to the Gröbner basis computation since the first three might be insufficient for extracting the common root. Finally, we were able to obtain the correct values for $p_1 + q_1$ and $p_2 + q_2$, which factorized N_1 and N_2, respectively.

In Table 4, the achieving rate estimates the attack performance in our lattice settings with different dimensions. The respective average achieving rates for $m = 20,\ 50,\ 105$ are 76.5%, 82.8%, 86.3%, which implies that a lattice with higher dimension indeed leads to better attack performance. Thus, we see that the experimental bound is a few bits away from the theoretical one since the lattice dimension is limited. The lattice dimension is a crucial factor in our attacks and it has a significant impact on the attack performance since we assume a large lattice dimension in the theoretical analysis.

5.3. Experimental results for Proposition 4

For our special implicit-key attack on RSA with the common private key d and N_1, N_2 of the same bit-size l, we generated two l-bit (i.e., $l = 1024,\ 2048,\ 3072$) RSA moduli and two public exponents (denoted by α_1, α_2 and $\alpha = \alpha_1 + \alpha_2$) appeared in the experiments were nearly of full bit-size. We omitted β_1 and β_2 as there is no implicit information about private keys. Moreover, we applied various lattice constructions indicated by s and $t = \tau s$. Table 5 displays the comparison of the insecure bounds. The theoretical upper bound on δ for specific parameters in our simplified strategy is provided by the δ_t-column. The experimental upper bound on δ for specific parameters in our computer experiments is provided by the δ_e-column. The AR-column indicates the achieving rate calculated as δ_e/δ_t, which compares our experimental bound with the theoretical one. We use m to represent the matching

Table 3
The comparison of theoretical and experimental results on δ for Proposition 2.

l	α_1	α_2	α	β_1	β_2	β	δ_t	δ_e	AR	m	Time
1024	0.998	0.999	1.997	0.010	0.010	0.020	0.255	0.252	98.8%	56	53.463 s
1024	1.000	0.998	1.998	0.040	0.040	0.080	0.270	0.267	98.8%	56	54.895 s
1024	0.999	0.998	1.997	0.060	0.100	0.160	0.290	0.287	98.9%	56	56.475 s
1024	0.997	0.999	1.996	0.100	0.140	0.240	0.310	0.308	99.3%	56	47.419 s
1024	0.999	0.999	1.998	0.156	0.151	0.307	0.327	0.325	99.3%	56	37.093 s
2048	0.999	0.999	1.998	0.012	0.015	0.027	0.257	0.253	98.4%	56	218.083 s
2048	1.000	0.999	1.999	0.061	0.100	0.161	0.290	0.286	98.6%	56	248.810 s
2048	0.999	0.999	1.998	0.156	0.151	0.307	0.327	0.324	99.0%	56	169.537 s
3072	1.000	1.000	2.000	0.012	0.015	0.027	0.257	0.253	98.4%	56	520.627 s
3072	1.000	1.000	2.000	0.100	0.141	0.241	0.310	0.305	98.3%	56	586.772 s
3072	0.999	1.000	1.999	0.149	0.150	0.299	0.325	0.322	99.0%	56	509.864 s

Table 4
The comparison of theoretical and experimental results on δ for Proposition 3.

l	α_1	α_2	α	β_1	β_2	β	δ_t	δ_e	AR	m	Time
1024	0.998	0.999	1.997	0.015	0.012	0.027	0.425	0.363	85.4%	50	21.412 s
1024	0.998	1.000	1.998	0.049	0.068	0.117	0.456	0.382	83.7%	50	21.319 s
1024	0.997	0.999	1.996	0.088	0.088	0.176	0.475	0.393	82.7%	50	20.843 s
1024	0.998	0.999	1.997	0.098	0.117	0.215	0.488	0.399	81.7%	50	21.684 s
1024	1.000	1.000	2.000	0.146	0.176	0.322	0.524	0.422	80.5%	50	23.654 s
2048	1.000	1.000	2.000	0.044	0.051	0.095	0.448	0.387	86.3%	105	3578.821 s
2048	0.999	0.999	1.998	0.098	0.117	0.215	0.488	0.425	87.0%	105	3787.419 s
2048	1.000	0.999	1.999	0.146	0.176	0.322	0.524	0.449	85.6%	105	4016.093 s
3072	0.999	0.998	1.997	0.044	0.051	0.095	0.448	0.358	79.9%	20	4.360 s
3072	1.000	0.999	1.999	0.098	0.117	0.215	0.488	0.371	76.0%	20	4.662 s
3072	0.998	0.999	1.997	0.146	0.176	0.322	0.524	0.386	73.6%	20	4.859 s

Table 5
The comparison of theoretical and experimental results on δ for Proposition 4.

l	α_1	α_2	α	δ_t	δ_e	AR	s	t	τ	m	Time
1024	0.999	0.999	1.998	0.334	0.330	98.8%	2	0	0	15	0.429 s
1024	0.999	0.997	1.996	0.335	0.331	98.8%	3	0	0	35	8.073 s
1024	0.998	0.997	1.995	0.358	0.333	93.0%	2	1	0.500	41	9.567 s
1024	0.996	0.996	1.992	0.336	0.332	98.8%	4	0	0	70	157.614 s
1024	1.000	0.998	1.998	0.403	0.354	87.8%	3	1	0.333	85	276.154 s
2048	1.000	0.998	1.998	0.329	0.324	98.4%	3	0	0	35	30.004 s
2048	1.000	1.000	2.000	0.357	0.334	93.5%	2	1	0.500	41	35.810 s
2048	1.000	1.000	2.000	0.402	0.356	88.5%	3	1	0.333	85	745.443 s
3072	1.000	1.000	2.000	0.333	0.331	99.3%	2	0	0	15	2.614 s
3072	1.000	0.999	1.999	0.357	0.333	93.2%	2	1	0.500	41	65.679 s
3072	1.000	0.999	1.999	0.402	0.356	88.5%	3	1	0.333	85	1644.579 s

lattice dimension, which is determined by s, t and τ. We use Time to denote the running time (measured in seconds).

We gathered enough polynomials with a common root over the integers for each experiment. More polynomials were added to the Gröbner basis computation since the first three might be insufficient for extracting the common root. Finally, we were able to obtain the correct values of $p+q$, which directly led to factorization of N. From Table 5, we observe that the experimental results get higher when the lattice dimension increases. Additionally, the average achieving rate is 98.8% for $t = 0$ but it is 90.7% for $t = 1$. This phenomenon implies that we need greater s for non-zero t in order to achieve better attack performance.

5.4. Experimental results for Proposition 5

For our extended implicit-key attack when given $n = 2$ different bit blocks in the middle of the private keys, we generated two l-bit (i.e., $l = 1024, 2048, 3072$) RSA moduli and two public exponents (denoted by α_1, α_2 and $\alpha = \alpha_1 + \alpha_2$) appeared in the experiments were nearly of full bit-size. The implicit information of respective shared MSBs, middle bits and LSBs were indicated by $\beta_1, \beta_2, \beta_3$ and $\beta = \beta_1 + \beta_2 + \beta_3$. Because the lattice construction was fixed by $s = 3$, $t = 0$, we needed to reduce an 84-dimension lattice. Table 6 displays the comparison of experimental insecure bounds. The theoretical upper bound on δ for specific parameters in our simplified strategy is provided by the δ_t-column. The experimental one for specific parameters in our computer experiments is provided by the δ_e-column. The AR-column indicates the achieving rate calculated as δ_e/δ_t, which compares our experimental bound with the theoretical one. We use m to represent the matching lattice dimension, and Time to denote the running time (measured in seconds).

We gathered enough polynomials with a common root over the integers for each experiment. More polynomials were added to the Gröbner basis computation since the first three might be insufficient for extracting the common root. Finally, we were able to obtain the correct values for p_1+q_1 and p_2+q_2, which factorized N_1 and N_2, respectively. We established the value of x_3 before determining the solution to the other variables if the Gröbner basis computation did not immediately produce the desired root. According to Table 6, $m = 84$ is sufficient since the experimental bound is so close to the theoretical one, which is based on the observation that the average achieving rate is 98.3%.

6. Conclusion

In this study, we review a novel RSA attack scenario with implicitly correlated private keys and make further extensions. Our goal is to

Table 6
The comparison of theoretical and experimental results on δ for Proposition 5.

l	α_1	α_2	α	β_1	β_2	β_3	β	δ_t	δ_e	AR	m	Time
1024	1.000	0.997	1.997	0.010	0.010	0.020	0.040	0.260	0.256	98.4%	84	473.531 s
1024	0.999	0.999	1.998	0.049	0.049	0.049	0.147	0.287	0.283	98.6%	84	352.497 s
1024	0.999	0.999	1.998	0.039	0.059	0.098	0.196	0.299	0.293	97.9%	84	337.486 s
1024	1.000	0.999	1.999	0.078	0.098	0.078	0.254	0.314	0.311	99.0%	84	248.021 s
1024	0.999	1.000	1.999	0.127	0.107	0.059	0.293	0.323	0.320	99.0%	84	186.509 s
2048	1.000	0.998	1.998	0.039	0.098	0.059	0.196	0.299	0.293	97.9%	84	1653.496 s
2048	0.999	0.999	1.998	0.078	0.098	0.078	0.254	0.314	0.310	98.7%	84	1417.193 s
2048	0.999	1.000	1.999	0.127	0.107	0.059	0.293	0.323	0.319	98.7%	84	1037.386 s
3072	0.999	0.998	1.997	0.039	0.098	0.059	0.196	0.299	0.291	97.3%	84	3179.607 s
3072	0.999	0.999	1.998	0.078	0.098	0.078	0.254	0.314	0.308	98.0%	84	2756.840 s
3072	1.000	0.999	1.999	0.127	0.107	0.059	0.293	0.323	0.317	98.1%	84	2147.563 s

factor given RSA moduli using implicit knowledge of the private keys that is already known. Such implicit knowledge specifically alludes to the known numbers of unknown common bits distributed among unknown correlated keys. With the help of the lattice-based method, which is adapted for solving integer polynomial equations, we present basic and extended implicit-key attacks.

The validity of our proposed attacks is verified and we reveal the vulnerability of RSA using implicitly correlated keys. Moreover, our work covers two special cases of a common modulus or a common private key. The corresponding results are stronger than previous ones. The experiments confirm the correctness and efficiency of the proposed attacks. We are able to launch a successful implicit-key attack and factor given RSA moduli in a matter of seconds.

CRediT authorship contribution statement

Mengce Zheng: Conceptualization, Methodology, Investigation, Software, Validation Writing – original draft, Writing – review & editing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The author would like to thank the anonymous reviewers for their detailed comments and useful suggestions, which improved this paper in terms of both technical and editorial quality. This work was supported by the National Natural Science Foundation of China, grant number 62002335, Ningbo Natural Science Foundation, grant number 2021J174 and Ningbo Young Science and Technology Talent Cultivation Program, grant number 2023QL007.

References

[1] Rivest RL, Shamir A, Adleman LM. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 1978;21(2):120–6. http://dx.doi.org/10.1145/359340.359342.

[2] Boneh D. Twenty years of attacks on the RSA cryptosystem. Notices Amer Math Soc 1999;46(2):203–13.

[3] May A. Using LLL-reduction for solving RSA and factorization problems. In: Nguyen PQ, Vallée B, editors. The LLL algorithm - Survey and applications, information security and cryptography. Springer; 2010, p. 315–48. http://dx.doi.org/10.1007/978-3-642-02295-1_10.

[4] Coppersmith D. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J Cryptol 1997;10(4):233–60. http://dx.doi.org/10.1007/s001459900030.

[5] Boneh D, Durfee G. Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans Inf Theory 2000;46(4):1339–49. http://dx.doi.org/10.1109/18.850673.

[6] Blömer J, May A. New partial key exposure attacks on RSA. In: Boneh D, editor. Advances in cryptology - CRYPTO 2003, 23rd annual international cryptology conference, Santa Barbara, California, USA, August (2003) 17-21, proceedings. Lecture notes in computer science, vol. 2729, Springer; 2003, p. 27–43. http://dx.doi.org/10.1007/978-3-540-45146-4_2.

[7] Coron J. Finding small roots of bivariate integer polynomial equations revisited. In: Cachin C, Camenisch J, editors. Advances in cryptology - EUROCRYPT 2004, international conference on the theory and applications of cryptographic techniques, Interlaken, Switzerland, May (2004) 2-6, proceedings. Lecture notes in computer science, vol. 3027, Springer; 2004, p. 492–505. http://dx.doi.org/10.1007/978-3-540-24676-3_29.

[8] Ernst M, Jochemsz E, May A, de Weger B. Partial key exposure attacks on RSA up to full size exponents. In: Cramer R, editor. Advances in cryptology - EUROCRYPT 2005, 24th annual international conference on the theory and applications of cryptographic techniques, Aarhus, Denmark, May (2005) 22-26, proceedings. Lecture notes in computer science, vol. 3494, Springer; 2005, p. 371–86. http://dx.doi.org/10.1007/11426639_22.

[9] Jochemsz E, May A. A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In: Lai X, Chen K, editors. Advances in cryptology - ASIACRYPT 2006, 12th international conference on the theory and application of cryptology and information security, Shanghai, China, December (2006) 3-7, proceedings. Lecture notes in computer science, vol. 4284, Springer; 2006, p. 267–82. http://dx.doi.org/10.1007/11935230_18.

[10] Coron J. Finding small roots of bivariate integer polynomial equations: A direct approach. In: Menezes A, editor. Advances in cryptology - CRYPTO 2007, 27th annual international cryptology conference, Santa Barbara, CA, USA, August (2007) 19-23, proceedings. Lecture notes in computer science, vol. 4622, Springer; 2007, p. 379–94. http://dx.doi.org/10.1007/978-3-540-74143-5_21.

[11] Sarkar S, Maitra S. Approximate integer common divisor problem relates to implicit factorization. IEEE Trans Inf Theory 2011;57(6):4002–13. http://dx.doi.org/10.1109/TIT.2011.2137270.

[12] Takayasu A, Kunihiro N. Partial key exposure attacks on RSA: achieving the Boneh-Durfee bound. Theoret Comput Sci 2019;761:51–77. http://dx.doi.org/10.1016/j.tcs.2018.08.021.

[13] Suzuki K, Takayasu A, Kunihiro N. Extended partial key exposure attacks on RSA: Improvement up to full size decryption exponents. Theoret Comput Sci 2020;841:62–83. http://dx.doi.org/10.1016/j.tcs.2020.07.004.

[14] May A, Nowakowski J, Sarkar S. Partial key exposure attack on short secret exponent CRT-RSA. In: Tibouchi M, Wang H, editors. Advances in cryptology - ASIACRYPT 2021-27th international conference on the theory and application of cryptology and information security, Singapore, December (2021) 6-10, proceedings, part I. Lecture notes in computer science, vol. 13090, Springer; 2021, p. 99–129. http://dx.doi.org/10.1007/978-3-030-92062-3_4.

[15] Wang S, Qu L, Li C, Fu S, Chen H. Finding small solutions of the equation bx-ay=z and its applications to cryptanalysis of the RSA cryptosystem. Adv Math Commun 2021;15(3):441–69. http://dx.doi.org/10.3934/amc.2020076.

[16] Lenstra AK, Lenstra HW, Lovász L. Factoring polynomials with rational coefficients. Math Ann 1982;261(4):515–34.

[17] Boneh D, Durfee G, Frankel Y. An attack on RSA given a small fraction of the private key bits. In: Ohta K, Pei D, editors. Advances in cryptology - ASIACRYPT '98, international conference on the theory and applications of cryptology and information security, Beijing, China, October (1998) 18-22, proceedings. Lecture notes in computer science, vol. 1514, Springer; 1998, p. 25–34. http://dx.doi.org/10.1007/3-540-49649-1_3.

[18] May A, Ritzenhofen M. Implicit factoring: On polynomial time factoring given only an implicit hint. In: Jarecki S, Tsudik G, editors. Public key cryptography - PKC 2009, 12th international conference on practice and theory in public key cryptography, Irvine, CA, USA, March (2009) 18-20. proceedings. Lecture notes in computer science, vol. 5443, Springer; 2009, p. 1–14. http://dx.doi.org/10.1007/978-3-642-00468-1_1.

[19] Faugère J, Marinier R, Renault G. Implicit factoring with shared most significant and middle bits. In: Nguyen PQ, Pointcheval D, editors. Public key cryptography - PKC 2010, 13th international conference on practice and theory in public key cryptography, Paris, France, May (2010) 26-28. proceedings. Lecture notes in computer science, vol. 6056, Springer; 2010, p. 70–87. http://dx.doi.org/10.1007/978-3-642-13013-7_5.

[20] Lu Y, Peng L, Zhang R, Hu L, Lin D. Towards optimal bounds for implicit factorization problem. In: Dunkelman O, Keliher L, editors. Selected areas in cryptography - SAC 2015-22nd international conference, Sackville, NB, Canada, August (2015) 12-14, revised selected papers. Lecture notes in computer science, vol. 9566, Springer; 2015, p. 462–76. http://dx.doi.org/10.1007/978-3-319-31301-6_26.

[21] Zheng M, Hu H. Implicit-key attack on the RSA cryptosystem. In: Liu F, Xu J, Xu S, Yung M, editors. Science of cyber security - second international conference, SciSec 2019, Nanjing, China, August (2019) 9-11, revised selected papers. Lecture notes in computer science, vol. 11933, Springer; 2019, p. 354–62. http://dx.doi.org/10.1007/978-3-030-34637-9_26.

[22] Kulow A, Schamberger T, Tebelmann L, Sigl G. Finding the needle in the haystack: Metrics for best trace selection in unsupervised side-channel attacks on blinded RSA. IEEE Trans Inf Forensics Secur 2021;16:3254–68. http://dx.doi.org/10.1109/TIFS.2021.3074884.

[23] Aljuffri A, Zwalua M, Reinbrecht CRW, Hamdioui S, Taouil M. Applying thermal side-channel attacks on asymmetric cryptography. IEEE Trans Very Large Scale Integr Syst 2021;29(11):1930–42. http://dx.doi.org/10.1109/TVLSI.2021.3111407.

[24] Sarkar S, Maitra S. Cryptanalysis of RSA with two decryption exponents. Inf Process Lett 2010;110(5):178–81. http://dx.doi.org/10.1016/j.ipl.2009.11.016.

[25] Sarkar S, Maitra S. Cryptanalysis of RSA with more than one decryption exponent. Inf Process Lett 2010;110(8–9):336–40. http://dx.doi.org/10.1016/j.ipl.2010.02.016.

[26] Hinek MJ. On the security of some variants of RSA [Ph.D. thesis], University of Waterloo; 2007.

[27] Sun H, Wu M, Ting W, Hinek MJ. Dual RSA and its security analysis. IEEE Trans Inf Theory 2007;53(8):2922–33. http://dx.doi.org/10.1109/TIT.2007.901248.

[28] Markelova AV. Embedding asymmetric backdoors into the RSA key generator. J Comput Virol Hacking Tech 2021;17(1):37–46.

[29] Nemec M, Sýs M, Svenda P, Klinec D, Matyas V. The return of Coppersmith's attack: Practical factorization of widely used RSA moduli. In: Thuraisingham B, Evans D, Malkin T, Xu D, editors. Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. ACM; 2017, p. 1631–48. http://dx.doi.org/10.1145/3133956.3133969.

[30] Heninger N. RSA, DH, and DSA in the wild. In: Bos J, Stam M, editors. Computational cryptography: Algorithmic aspects of cryptology. London mathematical society lecture note series, Cambridge University Press; 2021, p. 140–81.

[31] Zheng M, Kunihiro N, Hu H. Lattice-based cryptanalysis of RSA with implicitly related keys. IEICE Trans Fundam Electron Commun Comput Sci 2020;103-A(8):959–68. http://dx.doi.org/10.1587/transfun.2019EAP1170.

[32] Jochemsz E. Cryptanalysis of RSA variants using small roots of polynomials [Ph.D. thesis], Technische Universiteit Eindhoven; 2007.

[33] Herrmann M, May A. Solving linear equations modulo divisors: On factoring given any bits. In: Pieprzyk J, editor. Advances in cryptology - ASIACRYPT 2008, 14th international conference on the theory and application of cryptology and information security, Melbourne, Australia, December (2008) 7-11. proceedings. Lecture notes in computer science, vol. 5350, Springer; 2008, p. 406–24. http://dx.doi.org/10.1007/978-3-540-89255-7_25.

[34] May A. New RSA vulnerabilities using lattice reduction methods [Ph.D. thesis], University of Paderborn; 2003, URL http://ubdata.uni-paderborn.de/ediss/17/2003/may/disserta.pdf.

[35] Howgrave-Graham N. Finding small roots of univariate modular equations revisited. In: Darnell M, editor. Cryptography and Coding, 6th IMA international conference, Cirencester, UK, December (1997) 17-19, proceedings. Lecture notes in computer science, vol. 1355, Springer; 1997, p. 131–42. http://dx.doi.org/10.1007/BFb0024458.

[36] The Sage Developers. SageMath the sage mathematics software system (Version 9.0). 2021, https://www.sagemath.org.