

Practical Attacks on Small Private Exponent RSA: New Records and New Insights

Qiang Li¹, Qun-xiong Zheng^{1*} and Wen-feng Qi^{1†}

^{1*}PLA Strategic Support Force Information Engineering
University, 62 Kexue Road, Zhengzhou, 450001, P.R. China.

*Corresponding author(s). E-mail(s): qunxiong_zheng@163.com;
Contributing authors: hnuliqiang@163.com; wenfeng.qi@263.net;

[†]These authors contributed equally to this work.

Abstract

As a typical representative of the public key cryptosystem, RSA has attracted a great deal of cryptanalysis since its invention, among which a famous attack is the small private exponent attack. It is well-known that the best theoretical upper bound for the private exponent d that can be attacked is $d \leq N^{0.292}$, where N is a RSA modulus. However, this bound may not be achieved in practical attacks since the lattice constructed by Coppersmith method may have a large enough dimension and the lattice-based reduction algorithms cannot work so well in both efficiency and quality. In this paper, we propose a new practical attack based on the binary search for the most significant bits (MSBs) of prime divisors of N and the Herrmann-May's attack in 2010. The idea of binary search is inspired by the discovery of phenomena called "multivalued-continuous phenomena", which can significantly accelerate our attack. Together with several carefully selected parameters according to our exact and effective numerical estimations, we can improve the upper bound of d that can be practically achieved. We believe our method can provide some inspiration to practical attacks on RSA with mainstream-size moduli.

Keywords: Practical attack, Small private exponent attack, MSBs guess, Multivalued-continuous phenomena, Binary search

1 Introduction

1.1 Background

Since proposed in 1978, RSA public key cryptosystem [1] plays an important role in lots of fields such as data encryption, key encapsulation, etc. A variety of security analysis come along with its widespread application. As is known to all, the small private exponent attack is one kind of the most famous attacks on RSA.

For the sake of efficiency in the decryption process, one may choose a small private exponent d relative to the RSA modulus N when generating the parameters. Unfortunately, in 1990, Wiener [2] showed that the RSA cryptosystem was insecure once $d \leq \frac{1}{3}N^{1/4}$ using a continued fraction method. Many attempts were made based on Wiener's idea, $N^{1/4}$ was still the order of magnitude of the upper bound of d without any additional condition, though multiplied by a better coefficient [3–5]. A virtual significant improvement was made in [6] by Boneh and Durfee with the lattice-based strategy using the Coppersmith method. They firstly constructed a triangular lattice and got a bound $d \leq N^{0.284}$. By removing some unhelpful polynomials, they obtained a sublattice from the original one and improved the bound to $d \leq N^{0.292}$. However, the non-square sublattice brought plenty of troubles in the computation of the lattice determinant. In 2010, Herrmann and May [7] (we call it HM2010 attack for short below) applied the technique of unravelled linearization, which is introduced by Herrmann and May [8], to attack RSA and achieved the same bound as [6]. More importantly, the lattice constructed in [7] is a lower triangular square matrix which can avoid the complicated computation in [6]. Some generalizations of Boneh-Durfee's result can be seen in [9, 10], but the upper bound of d was no better than [6] indeed. Overall, the upper bound $d \leq N^{0.292}$ showed by Boneh and Durfee remains to be the best theoretical achievement till now.

1.2 Previous works

We note that all the bounds of the private exponent above are theoretical, i.e., those bounds are asymptotic which means that they will be achieved only if the lattice dimensions are large enough. As we know, the reduction algorithm cannot work so well in both efficiency and quality in a high-dimension lattice, which means that the best theoretical upper bound given in [6, 7] may not be achieved in practice. A natural question is what practical bound can be achieved. Therefore, many attempts have been made hoping to give an answer to this question.

One kind of the implementation of the practical small private exponent attacks is using the lattices constructed in [6, 7]. In [11], Boneh and Durfee ran their experiments to attack the RSA cryptosystem successfully when $d \leq N^{0.265}$ with moduli of 1000 bits, 3000 bits and 10000 bits. Later, Durfee got a better results, namely, $d \leq N^{0.277}$ for a 1024-bit-modulus RSA and $d \leq N^{0.275}$

for a 2048-bit-modulus RSA [21]. In 2001, Blömer and May proposed a new attack and carried out some experiments [12]. In comparison with [11], they didn't improve the asymptotic bound but the dimension of the lattice they constructed was lower under the same bound of d and the same size of N . For a 1000-bit modulus N , they can implement an effective attack in 6 days as long as $d \leq N^{0.278}$. Without the additional tricks like [6] (such as using a reduction variant, Chebychev polynomials or some guess strategy), the early lattice-based practical attacks [11, 12, 21] as well as the attack in [13] didn't seem to break through the bound $d \leq N^{0.278}$ when the size of N is not less than 1024 bits. In 2021, Miller et al. [14] carried out their "focus group" attack in which the bound of d was improved to $d \leq N^{0.280}$ for a 1000-bit modulus N .

Another kind of RSA practical attacks aimed to improve the bound of d is implemented with knowledge of some bits of a prime factor p of N . Early attacks of this kind were mainly theoretical since a large continuous fragment of bits must be known [23, 24], which may not be feasible in practice. In 2003, Suk stated that by knowing just $\frac{1}{100} \log_2 N$ most significant bits (MSBs) of p , one could break the RSA cryptosystem for $d < N^{0.30}$ [22]. However, in his experiments, by knowing 10 MSBs of p , he got a bound $d \leq N^{0.285}$ for a 1000-bit modulus N , which did not reach $N^{0.30}$ as he stated (detailed results can be seen in Table 5.8 in [22]). Later in 2008, Sarkar et al. ran large numbers of experiments and got some more detailed corresponding tables like [22] by searching exhaustively a few MSBs of p [25]. The tables highlight that the small private exponent RSA can be successfully attacked with a low-dimension lattice in practice. Note that the experiments in [22] and [25] are confirmatory, i.e., they verified that RSA can be broken when a few MSBs of p were already known, instead of searching for each candidate of their values. They did not finish the the complete attack and the running time of the the complete attack was estimated. For example, in [25] they stated that, to break a 1000-bit-modulus RSA for $d \leq N^{0.285}$, the total time was $2^{15} \times 484$ seconds since each run required around 484 seconds and 15 MSBs were needed to search, which means that they needed about a week with a cluster of 26 machines using a 48-dimension lattice. More results of this kind can be seen in [26–30].

1.3 Our contributions

In this paper, we focus on the practical attack on RSA. As we can see from the previous works, there is still a considerable gap between the best theoretical bound of d and the practical one. Then, can the gap be further narrowed? If so, what can be done to achieve this goal? With these questions, we firstly take 1024-bit-modulus RSA as an example to carry out our practical attack and then expand to the practical attack of RSA with other moduli, such as 2048-bit-modulus RSA.

Firstly, we do a complete, detailed calculation for specific values of the dimension and the exponents of the variables in the lattice determinant. Values of parameters in the lattice determinant were usually given by orders of magnitude in previous works. However, when the dimension

of the lattice is not large enough, those values may have great deviation from real values of the parameters. Therefore, in practical attacks, extremely specific parameter values are quite essential in instructional estimations and attack implementations. In order to reduce the errors, by retaining all the low order terms in the calculations, we give accurate analytical approximations and exact numerical values of the dimension and the exponents.

Secondly, we give a detailed and relatively accurate numerical estimation for the bound of d . And then, guided by the estimation, we achieve an upper bound $d \leq N^{0.285}$ for a 1024-bit-modulus N , which is the best upper bound of this kind of attack as far as we know. For the parameters m and t of Coppersmith method (for detailed introduction, see section 2.2 and section 3), we get the optimal value of t responding to every value of m by using our analytical approximations. After that, based on the specific numerical parameter values and the experimental performance estimation of LLL algorithm provided by Nguyen and Stehlé [15], we give a detailed and relatively accurate estimation for the bound of d . The estimation table for the solvable upper bound of d we make is well consistent with both our experimental results and those in [25] especially in the case of medium-dimension lattices, which is mainly used in our practical attack. Guided by our estimation table, we implement our first practical attack to find the upper bound we can achieve based on the HM2010's lattice. When the parameters m and t are set to 25 and 10, within 17 days on our PC, we successfully attack 1024-bit-modulus RSA for $d \leq N^{0.285}$ (without any exhaustion strategy or additional side channel information), which is better than all the previous results we know.

Thirdly, inspired by the multivalued-continuous phenomena in our experiments, we propose a new effective practical attack based on the binary search for some MSBs of p . Our experiments imply that it seems very difficult to successfully attack RSA for $d > N^{0.285}$ in practice if no other tricks are added, since the running time of the attack would be unacceptable. Then what can we do to further improve the practical bound of d ? A natural idea may be to enumerate all possible values of several MSBs of p , which had been tried in some previous works. Frankly speaking, this idea is simple and trivial. However, during our implementations, we find some non-trivial and inspiring phenomena, which we call it the “multivalued-continuous phenomena” for convenience: (I) besides the real value of the MSBs of p and q , much more additional exhaustive values (helpful guess values) can also help to attack the RSA cryptosystem successfully; (II) the helpful guess values appear continuously around real values of the MSBs of p and q ; (III) the closer p and q are, the more helpful guess values there will be. Based on these inspiring phenomena, we propose a new practical attack in which one helpful guess value can be efficiently found by the binary search. As a result, we can significantly accelerate the attack, which is far better than other current practical results. More precisely, for a relatively close p and q , e.g., they share 4 MSBs, a 1024-bit-modulus RSA can be successfully attacked within several hours in a single

PC for $d \leq N^{0.292}$; even when p is quite far from q , the overwhelming majority of 1024-bit-modulus RSA can be broken in a month with a single PC for $d \leq N^{0.292}$. By the way, the expression “ p is close to q ” means the value of $|p - q|$ is relatively small, while “ p is far from q ” means the value of $|p - q|$ is relatively large.

Finally, we implement our new practical attack on 2048-bit-modulus RSA and also get a nice upper bound. For a relatively close p and q in our experiment (p and q share about 3 MSBs), the RSA can be successfully attacked in about a week with a single PC for $d \leq N^{0.287}$; when p is quite far from q , the overwhelming majority of 2048-bit-modulus RSA can be broken in about a month with a single PC for $d \leq N^{0.287}$. Moreover, for a quite close p and q , the RSA also can be efficiently broken for $d \leq N^{0.292}$, e.g., in our experiment, with p and q sharing 50 MSBs, the RSA can be successfully attacked within 12 days with a single PC for $d \leq N^{0.292}$. Though the promotion effect is not so good as that of attack on 1024-bit-modulus RSA, the bound $d \leq N^{0.287}$ (our attack on RSA can succeed with very special p and q for $d \leq N^{0.292}$ but may not work well in a general case) is still better than all previous works as far as we know.

We would like to note that all our experimental results are obtained with a single PC, for more computing power, the running time given by our attack will surely be further improved.

1.4 Organization of the paper

The rest of this paper is organized as follows. In Section 2, we recall some preliminary knowledge and lemmas to be used latter. In Section 3, we revisit the Herrmann and May’s attack in 2010. In Section 4, we introduce in detail our new practical attack on RSA based on the binary search. In Section 5, we show some results of our experiments. Section 6 is the conclusion.

2 Preliminaries

2.1 Lattices and the LLL algorithm

Let $\mathbf{u}_1, \dots, \mathbf{u}_w \in \mathbb{R}^n$ be w linearly independent vectors with $w \leq n$. The n -dimensional lattice L , spanned by $\mathbf{u}_1, \dots, \mathbf{u}_w$, is the set of all integer linear combinations of $\mathbf{u}_1, \dots, \mathbf{u}_w$, namely,

$$L = \left\{ \sum_{i=1}^w k_i \mathbf{u}_i \mid k_1, \dots, k_w \in \mathbb{Z} \right\}.$$

The set of vectors $\mathbf{u}_1, \dots, \mathbf{u}_w$ is called a basis of L and the integer w is called the rank of L . Specially, the lattice L is called full rank if $w = n$. Let \mathbf{U} be the $w \times n$ matrix consisting of row vectors $\mathbf{u}_1, \dots, \mathbf{u}_w$. Then the determinant of L is defined by $\det L = \sqrt{|\mathbf{U}\mathbf{U}^t|}$. A famous hard problem is to find a short non-zero vector in L , especially the shortest one. The famous LLL algorithm [18]

behaves well in finding a relatively short and nearly orthogonal lattice basis in polynomial time. Specifically, the properties of the output lattice basis by LLL algorithm can be seen in the following lemma.

Lemma 1 Let L be a full-rank lattice of dimension w and let $\tilde{\mathbf{v}}_1, \dots, \tilde{\mathbf{v}}_w$ be a reduced basis of L output by the LLL algorithm. Then

$$\|\tilde{\mathbf{v}}_i\| \leq 2^{\frac{w(w-1)}{4(w+1-i)}} (\det L)^{\frac{1}{(w+1-i)}} \text{ for any } 1 \leq i \leq w,$$

where $\|\tilde{\mathbf{v}}_i\|$ is the Euclidean norm of $\tilde{\mathbf{v}}_i$.

In most cases, the first reduced basis vector $\tilde{\mathbf{v}}_1$ attracts major attention. However, the upper bound $2^{\frac{w-1}{4}} (\det L)^{\frac{1}{w}}$ of $\tilde{\mathbf{v}}_1$ given by Lemma 1 is quite rough. In order to give a precise estimation as an instruction for our practical attack, we need a more accurate bound for $\tilde{\mathbf{v}}_1$. An experimental estimation for the shortest lattice vector is given as follows by Nguyen et al. [15], namely,

$$\lambda_1(L)/(\det L)^{1/w} \approx (1.02)^w,$$

where $\lambda_1(L)$ denotes the Euclidean norm of the shortest lattice vector of L . This estimation will be of great help in our attack later.

2.2 Coppersmith method

Let $\|h(x_1, \dots, x_r)\|$ denote the norm of a polynomial $h(x_1, \dots, x_r)$ which represents the Euclidean norm of the coefficient vector. Consider a modular equation $h(x_1, \dots, x_r) = 0 \pmod{M}$, where all the absolute values of the target solutions x_1, \dots, x_r are bounded by X_1, \dots, X_r , respectively. In 1996, a polynomial time algorithm to find all the solutions under the boundary was given by Coppersmith if $\prod_{i=1}^r X_i$ is smaller than M , determinately when $r \leq 2$ and heuristically when $r > 2$ [32]. Later, an important work was done by Howgrave-Graham [17] who gave a simpler sufficient condition to transform a modular equation into an integer equation.

Lemma 2 (Howgrave-Graham) Let m, M, X_1, \dots, X_r be positive integers. Let $g(x_1, \dots, x_r) \in \mathbb{Z}[x_1, \dots, x_r]$ be a polynomial with at most n monomials and let $\|g(x_1, \dots, x_r)\|$ denote the norm of the polynomial $g(x_1, \dots, x_r)$. If

- (1) $g(\tilde{x}_1, \dots, \tilde{x}_r) = 0 \pmod{M^m}$, where $|\tilde{x}_1| < X_1, \dots, |\tilde{x}_r| < X_r$ and
- (2) $\|g(x_1 X_1, \dots, x_r X_r)\| < M^m / \sqrt{n}$,

then $g(\tilde{x}_1, \dots, \tilde{x}_r) = 0$ holds over \mathbb{Z} .

Lemma 2 provides a key instruction in RSA cryptanalysis. From Condition (2) of Lemma 2, we need to find polynomials with relatively small norms. Since there is one-to-one correspondence between a polynomial and its coefficient vector, the target of finding polynomials with small norms boils down to finding

short non-zero vectors in the constructed lattice, which can be achieved in polynomial time by LLL algorithm.

After finding enough short nonzero vectors, we need to compute the common roots of the polynomials corresponding to the vectors. Usually, we need the following heuristic assumption. Recall that polynomials $f_1, f_2, \dots, f_m \in k[x_1, x_2, \dots, x_n]$ are called algebraically independent over a field k , if there is no nonzero m -variate polynomial $\Phi \in k[y_1, y_2, \dots, y_m]$ such that $\Phi(f_1, f_2, \dots, f_m) = 0$.

Assumption 1 The polynomials output by the LLL algorithm are algebraically independent, and so the common roots of these polynomials can be computed by computing resultants or finding a Gröbner basis.

This assumption has been verified by experiments before as well as ours in Section 5.

3 The HM2010 attack revisited

Our practical attack on RSA in this paper relies heavily on the work of Herrmann and May. Therefore, in this section we will revisit the HM2010 attack briefly. In order to show its essence, we will revisit Boneh and Durfee's work together. For convenience, the lattices given by Boneh and Durfee that yield the bounds 0.284 and 0.292 are called BD-0.284-lattice and BD-0.292-lattice, respectively.

Let $N = pq$ be a public RSA modulus whose prime factors p and q are of the same bitsize. A public exponent e and a private exponent d satisfy $ed \equiv 1 \pmod{\varphi(N)}$, i.e.,

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

for some integer k . Let $A = N + 1$ and $s = -p - q$. Then it can be seen from (1) that

$$k(A + s) + 1 \equiv 0 \pmod{e}.$$

Denote $f(x, y) := x(A + y) + 1$. The original aim to recover d and factor N became to find small roots of the polynomial $f(x, y) \pmod{e}$. With a fixed positive integer m , a parameter t to be optimized and the definitions of the following polynomials (usually called x -shift polynomials and y -shift polynomials respectively)

$$\begin{aligned} g_{i,k}(x, y) &:= x^i f^k e^{m-k}, k = 0, \dots, m \text{ and } i = 0, \dots, m - k; \\ h_{j,k}(x, y) &:= y^j f^k e^{m-k}, k = 0, \dots, m \text{ and } j = 1, \dots, t, \end{aligned}$$

Boneh and Durfee constructed the BD-0.284-lattice using the Coppersmith method. All the used shift polynomials are ordered as

$$\begin{aligned} g_{i,k}(x, y) &\prec h_{j,k'}(x, y) \text{ for any } i, j, k, k', \\ g_{i,k}(x, y) &\prec g_{i',k'}(x, y) \text{ for } i + k < i' + k', \end{aligned}$$

$$g_{i,k}(x, y) \prec g_{i',k'}(x, y) \text{ for } i + k = i' + k' \text{ and } i > i',$$

$$h_{j,k}(x, y) \prec h_{j',k'}(x, y) \text{ for } j < j',$$

$$h_{j,k}(x, y) \prec h_{j,k'}(x, y) \text{ for } k < k'.$$

A simple BD-0.284-lattice with $m = 2, t = 1$ is shown below (9×9 matrix).

	1	x	xy	x^2	x^2y	x^2y^2	y	xy^2	x^2y^3
e^2	e^2								
xe^2	e^2X								
fe	e	eAX	eXY						
x^2e^2				e^2X^2					
xfe	eX		eAX^2		eX^2Y				
f^2	1	$2AX$	$2XY$	A^2X^2	$2AX^2Y$	X^2Y^2			
ye^2							e^2Y		
yfe	$eAXY$					eY	eXY^2		
yf^2	$2AXY$		A^2X^2Y		$2AX^2Y^2$	Y	$2XY^2$	X^2Y^3	

Using the BD-0.284-lattice together with $m \rightarrow \infty$, they got the bound $d \leq N^{0.284}$. In order to decrease the determinant of the lattice and improve the upper bound of d , they tried to remove some rows that enlarge the determinant. They constructed the BD-0.292-lattice by throwing away the y -shift polynomials $y^j f^k e^{m-k}$ from the BD-0.284-lattice for all j and $k < \lfloor m/t \rfloor j$. For example, when $m = 2$ and $t = 1$, the bold rows of the above BD-0.284-lattice is removed, and then the BD-0.292-lattice is as follows (7×9 matrix).

	1	x	xy	x^2	x^2y	x^2y^2	y	xy^2	x^2y^3
e^2	e^2								
xe^2	e^2X								
fe	e	eAX	eXY						
x^2e^2				e^2X^2					
xfe	eX		eAX^2		eX^2Y				
f^2	1	$2AX$	$2XY$	A^2X^2	$2AX^2Y$	X^2Y^2			
yf^2	$2AXY$		A^2X^2Y		$2AX^2Y^2$	Y	$2XY^2$	X^2Y^3	

With the BD-0.292-lattice, the upper bound of d is improved to $d \leq N^{0.292}$. However, the BD-0.292-lattice is not a square matrix which results in a complex computation of the lattice determinant.

In order to avoid the complex computation of the determinant, Herrmann and May applied a technique of unravelled linearization to construct a new square lattice and got the same bound $d \leq N^{0.292}$. They stated the reason why the BD-0.292-lattice is not square is that the first y -shift polynomial brings more than one new term to the lattice. This phenomenon results in more than one column adding to the lattice when one row is added. To solve the trouble, they applied the unravelled linearization technique to their lattice construction. In the technique the substitution $xy = u - 1$ is used twice. The first use changes $f(x, y)$ to $\tilde{f}(u, x) = u + Ax$. Since all the shift polynomials are added according to the order defined above, the second use can make sure that every new-added y -shift polynomial adds only one new term to the set constituted by all the terms of the former polynomials, which keeps the lattice being a square matrix. More detailed, consider $y^j \tilde{f}^k$ being added to the lattice (the factor e^{m-k} is omitted as it does not influence the set of terms). Since $\tilde{f}(u, x) = u + Ax$, it

follows that

$$y^j \tilde{f}^k = u^k y^j + \sum_{i=1}^k \binom{k}{i} A^i u^{k-i} x^i y^j.$$

The term $u^k y^j$ is a new-added term. Consider the other terms in $y^j \tilde{f}^k$. Using the second substitution $xy = u - 1$ one can get

$$u^{k-i} x^i y^j = u^{k-i} (u-1)^{\min\{i,j\}} x^{i-\min\{i,j\}} y^{j-\min\{i,j\}}.$$

If $i \geq j$, then

$$u^{k-i} x^i y^j = u^{k-i} (u-1)^j x^{i-j} = \sum_{l=0}^j \binom{j}{l} (-1)^{j-l} u^{k-i+l} x^{i-j},$$

whose terms already appear in the x -shift polynomials $x^{i-j} \tilde{f}^{k-i}, \dots, x^{i-j} \tilde{f}^{k-i+j}$; if $i < j$, then

$$u^{k-i} x^i y^j = u^{k-i} (u-1)^i y^{j-i} = \sum_{l=0}^i \binom{i}{l} (-1)^{i-l} u^{k-i+l} y^{j-i},$$

whose terms already appear in $y^{j-i} \tilde{f}^{k-i}, \dots, y^{j-i} \tilde{f}^k$. All these polynomials as well as the terms have been added to the lattice before. Therefore, by this technique, every y -shift polynomial $y^j \tilde{f}^k$ adds only one new term $u^k y^j$ to the lattice, which makes sure that the lattice is square. A simple HM2010 lattice with $m = 2, t = 1$ is shown below (7×7 matrix).

	1	x	u	x^2	xu	u^2	$u^2 y$
e^2	e^2						
xe^2		$e^2 X$					
$\tilde{f}e$		eAX	eU				
$x^2 e^2$				$e^2 X^2$			
$x \tilde{f}e$				eAX^2	eXU		
\tilde{f}^2				$A^2 X^2$	$2AXU$	U^2	
$y \tilde{f}^2$		$-A^2 X$	$-2AU$		$A^2 XU$	$2AU^2$	$U^2 Y$

Overall, the complete HM2010 attack can be briefly summarized as follows.

- (1) Use the substitution $xy = u - 1$, $f(x, y)$ becomes to $\tilde{f}(u, x) = u + Ax$, and then $g_{i,k}(x, y)$ and $h_{j,k}(x, y)$ are changed to $\tilde{g}_{i,k}(u, x)$ and $\tilde{h}_{j,k}(u, x, y)$.
- (2) Discard the y -shift polynomials $\tilde{h}_{j,k}(u, x, y)$ if $k < \lfloor m/t \rfloor j$, and then the retained y -shift polynomials are $\tilde{h}_{j,k}(u, x, y) := y^j \tilde{f}^k e^{m-k}$ with $j = 1, \dots, t$ and $k = \lfloor m/t \rfloor j, \dots, m$.
- (3) Reuse the substitution $xy = u - 1$ to substitute all the xy in the monomials of $\tilde{g}_{i,k}(u, x)$ and $\tilde{h}_{j,k}(u, x, y)$.
- (4) Construct a lower triangular lattice L based on $\tilde{g}_{i,k}(u, x)$ ($k = 0, \dots, m$ and $i = 0, \dots, m - k$) and $\tilde{h}_{j,k}(u, x, y)$ ($j = 1, \dots, t$ and $k = \lfloor m/t \rfloor j, \dots, m$).
- (5) Use the lattice basis reduction algorithm together with resultants computation or a Gröbner basis method and finally get the value of d .

The core point of this technique is the double use of the substitution $xy = u - 1$ in step (1) and (3). The first use greatly reduces the terms of the polynomials and results in a decrease of the lattice dimension. The second use makes sure that every

new-added y -shift polynomial adds only one new term to the set constituted by all the terms of the former polynomials, which keeps the lattice being a square matrix.

Let $d \leq N^\delta$ for some real number δ . Let $\tau = m/t$ and let s_X, s_Y, s_U, s_e denote the contribution of X, Y, U, e to the determinant $\det L$. Based on the simplified condition $\det L = X^{s_X} Y^{s_Y} U^{s_U} e^{s_e} \leq e^{m \dim L}$ from Lemma 2, it can be obtained that

$$\delta \cdot \frac{m^3}{6} + \frac{1}{2} \cdot \frac{\tau^2 m^3}{6} + \left(\delta + \frac{1}{2}\right) \cdot \frac{(1+2\tau)m^3}{6} + \frac{(2+\tau)m^3}{6} \leq \frac{(1+\tau)m^2}{2} \cdot m, \quad (2)$$

using the upper bounds $X = N^\delta, Y = N^{1/2}, U = N^{\delta+1/2}$ together with the approximate calculations of

$$\begin{aligned} s_X &= \frac{m^3}{6} + o(m^3), \\ s_Y &= \frac{\tau^2 m^3}{6} + o(m^3), \\ s_U &= \frac{(1+2\tau)m^3}{6} + o(m^3), \\ s_e &= \frac{(2+\tau)m^3}{6} + o(m^3), \\ \dim L &= \frac{(1+\tau)m^2}{2} + o(m^2). \end{aligned}$$

After getting an optimized value of $\tau = (1 - 2\delta)$, they finally successfully obtained the desired Boneh-Durfee bound

$$\delta \leq 1 - \frac{\sqrt{2}}{2} \approx 0.292.$$

For details, see [7].

4 A new practical small private exponent attack on RSA

The majority of small private exponent attacks on RSA mainly focus on the theoretical asymptotic upper bound of the private exponent d . Therefore, as far as we know, when calculating the values of the exponents in the lattice determinant, only the highest order terms of the parameter m are retained. Considering that the value of m cannot be too large in the practical attack, the low-order terms should be retained for accurate estimations. In this section, we will consider the modulus N with η -bit-size prime factors p and q where $q < p < 2q$ and set $d = N^\delta$.

With the same notations as Section 3, specific and relatively exact values of some parameters used in the calculations of the dimension $\dim L$ and the determinant $\det L$ are shown as follows.

Lemma 3 With the same notations as Section 3, we have

$$\begin{aligned} \dim L &= \sum_{k=0}^m \sum_{i=0}^{m-k} 1 + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m 1 \approx \frac{(m+1)(m+2)}{2} + \frac{\tau m^2 + (2\tau-1)m}{2}, \\ \det L &= X^{s_X} Y^{s_Y} U^{s_U} e^{s_e}, \end{aligned}$$

where

$$\begin{aligned}
 s_X &= \sum_{k=0}^m \sum_{i=0}^{m-k} i = \frac{m(m+1)(m+2)}{6}, \\
 s_Y &= \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m j \approx \frac{\tau^2 m^3 + 3\tau^2 m^2 + 3\tau m - m}{6}, \\
 s_U &= \sum_{k=0}^m \sum_{i=0}^{m-k} k + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m k \approx \frac{(4\tau^2 + 2\tau)m^3 + (9\tau^2 + 3\tau)m^2 + (7\tau - 1)m}{12\tau}, \\
 s_e &= \sum_{k=0}^m \sum_{i=0}^{m-k} (m-k) + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m (m-k) \\
 &\approx \frac{m(m+1)(m+2)}{3} + \frac{2\tau^2 m^3 + (3\tau^2 - 3\tau)m^2 - (3\tau - 1)m}{12\tau}.
 \end{aligned}$$

Proof. See Appendix A.

With the same consideration as above, the second condition in Lemma 2 cannot be simplified as $\det L \leq e^{m \dim L}$. Let $\tilde{\mathbf{v}}_1$ be the first reduced basis vector output by LLL algorithm. Strictly following Lemma 2, we need

$$\|\tilde{\mathbf{v}}_1\| < \frac{e^m}{\sqrt{\dim L}}.$$

By the LLL algorithm, the theoretical upper bound of $\tilde{\mathbf{v}}_1$ can be given by

$$\|\tilde{\mathbf{v}}_1\| < 2^{(\dim L - 1)/4} (\det L)^{1/\dim L}.$$

However, this estimation is too rough to instruct our experiments. Instead, in the practical attack, we adopt the experimental estimation of $\tilde{\mathbf{v}}_1$ given by Nguyen et al. in [15], i.e.,

$$\|\tilde{\mathbf{v}}_1\| < (1.02)^{\dim L} (\det L)^{1/\dim L}.$$

Then we can utilize the following modified estimation formula to obtain the upper bound of the solvable δ , namely,

$$(1.02)^{\dim L} (\det L)^{1/\dim L} < \frac{e^m}{\sqrt{\dim L}}. \quad (3)$$

In the rest of this section, we will at first explore the practical solvable bound of δ using HM2010's method. In order to break through the bound, we then try a few MSBs exhaustion of p to obtain new bounds of δ . At last, we introduce our new attack based on the binary search.

4.1 The practical solvable bound of δ in HM2010 attack

Firstly, we use the above method to estimate the upper bound of the solvable δ . Take the logarithm of both sides of Inequality (3) to base N , we can obtain

$$(\dim L)^2 \cdot \log_N 1.02 + \frac{1}{2} \dim L \cdot \log_N \det L + \log_N \det L < m \cdot \dim L \cdot \log_N e.$$

Since

$$\log_N 1.02 \approx \frac{\log_2 1.02}{2\eta}, \quad \log_N \dim L \approx \frac{1}{2\eta} \log_2 \dim L,$$

$$\log_N e \approx 1, \quad \log_N \det L \approx \delta \cdot s_X + \frac{1}{2}s_Y + \left(\delta + \frac{1}{2}\right) \cdot s_U + s_e,$$

it follows that

$$\delta < \frac{-(\dim L)^2 \cdot \frac{\log_2 1.02}{2\eta} + (m - \frac{\log_2 \dim L}{4\eta}) \cdot \dim L - \frac{1}{2}(s_Y + s_U) - s_e}{s_X + s_U}. \quad (4)$$

Since (4) is too complicated for us to give an optimal analytical expression of the bound, instead we show the discrete corresponding relation of the upper bound of δ with m and t . Using (4) together with analytical approximations in Lemma 3, we can get the optimal t corresponding to various m . Based on the parameters m and the optimal t , we can obtain the value of $1/\tau$ (namely, m/t), which can help us to get all the real values of the parameters in Lemma 3 by a rounding operation. Based on the real values and (4), a corresponding relation between the upper bound of δ and m, t is displayed in Table 1.

Table 1 A corresponding relation between δ and m, t for $2\eta = 1024$

m	Optimal t	Actual τ	δ	m	Optimal t	Actual τ	δ
5	2	0.4000	0.2679	16	7	0.4375	0.2821
6	2	0.3333	0.2704	17	7	0.4118	0.2827
7	3	0.4286	0.2731	18	7	0.3889	0.2832
8	3	0.3750	0.2752	19	8	0.4211	0.2834
9	4	0.4444	0.2763	20	8	0.4000	0.2839
10	4	0.4000	0.2779	21	9	0.4286	0.2840
11	5	0.4545	0.2789	22	9	0.4091	0.2844
12	5	0.4167	0.2797	23	10	0.4348	0.2845
13	5	0.3846	0.2806	24	10	0.4167	0.2848
14	6	0.4286	0.2811	25	10	0.4000	0.2851
15	6	0.4000	0.2818	26	11	0.4231	0.2854

As can be seen from Table 1, when $m = 21, t = 9$ and $\delta \leq 0.284$, the method of HM2010 can successfully recover the private exponent d . This is also consistent with our experimental results. Theoretical estimations and experimental results show that the minimum parameters selected for solving $\delta \leq 0.284$ are $m = 21, t = 9$, and the success rate of such experiments is more than 60% with 100 experiments. Similarly, if the private exponent d with the size $\delta \leq 0.285$ is required to be solved, the minimum parameters to be chosen are $m = 25, t = 10$. This is also consistent with our experiments since the rate of our experiments is about 60% (see Section 5 for details of our experiments). Considering the running time of our attack for $\delta \approx 0.285$, it seems very difficult to get a better bound of δ in practical attacks on RSA.

Remark 1 A slight mismatch may exist between the estimation in Table 1 and our experimental results if m is small. For example, when $m = 6$, the method of HM2010 can accomplish the attack with a high success rate as long as $\delta \leq 0.271$. When $m = 7$, using the method of HM2010 the bound can reach 0.274.

Remark 2 Inequality (4) has great deviation from asymptotic predictions when m is relatively large. For example, when $m = 76$, the predicted bound will reach the peak of 0.2875 under this estimation. When $m > 76$, the estimation bound will decline, which is inconsistent with the theoretical asymptotic result and the actual cognition. This experimental phenomenon may well illustrate the inapplicability of the approximation formula $(\lambda_1(L))/(\det L)^{1/\dim L} \approx (1.02)^{\dim L}$ in high dimensional case to some extent.

From Table 1 we can see that, to get a better bound of δ , we need a larger value of m , thus resulting in a lattice with a higher dimension. However, as we know, the LLL algorithm does not perform well in a lattice with a high dimension. This fact makes it unpractical to further improve the bound by utilizing HM2010's lattice merely. There is no doubt that new ideas need to be added into the attack in order to improve the practical bound of δ .

4.2 New bounds of δ with some MSBs exhaustion of p

When using HM2010 method directly to attack the small private exponent of RSA, there will inevitably be a bottleneck. That is, a larger value of m would lead to a larger practical upper bound of solvable δ ; while at the same time, when m increases, the dimension of the lattice increases rapidly, which results in a sharp rise in the running time. Based on our experiments, it seems very difficult to break 1024-bit-modulus RSA at a reasonable time with HM2010 attack if $\delta > 0.285$.

In order to further improve the practical upper bound of the assaultable private exponent, we look back to Inequality (3). We note that, when m and t are fixed, decreasing the value of $\det L$ will raise the upper bound of δ . In fact, let us consider the equation

$$\det L = X^{s_X} Y^{s_Y} U^{s_U} e^{s_e}.$$

It can be seen from Lemma 3 that s_X, s_Y, s_U, s_e will remain unchanged for fixed m and t . It is clear that if the bounds Y and U are decreased, then $\det L$ will decrease. A natural and trivial idea to achieve this goal is to try an exhaustive search for some MSBs of p , which is the same thought as Suk in [22] and Sarkar in [25]. At first we will show how to compute the MSBs of q (or p) based on the MSBs of p (or q).

Lemma 4 Let $N = pq$ and the η -bit-size prime factors p and q satisfy $q < p < 2q$. Let p_m, q_m be the s MSBs of p, q and let p_l, q_l be the $\eta - s$ least significant bits (LSBs) of p, q . Then we have

$$q_m = \left\lfloor \left\lfloor \frac{N}{p_m \cdot 2^{\eta-s} + 2^{\eta-s-1}} \right\rfloor / 2^{\eta-s} \right\rfloor + \alpha, \alpha \in \{-1, 0, 1\}, \quad (5)$$

$$p_m = \left\lfloor \left\lfloor \frac{N}{q_m \cdot 2^{\eta-s} + 2^{\eta-s-1}} \right\rfloor / 2^{\eta-s} \right\rfloor + \alpha', \alpha' \in \{-1, 0, 1\}. \quad (6)$$

Proof Let $\tilde{p} = p_m \cdot 2^{\eta-s} + 2^{\eta-s-1}$ and denote $\tilde{q} = N/\tilde{p}$. Then

$$|q - \tilde{q}| = \left| q - \frac{N}{\tilde{p}} \right| = \frac{q|p - \tilde{p}|}{\tilde{p}} = \frac{q|p_l - 2^{\eta-s-1}|}{\tilde{p}}.$$

The inequality $|p_l - 2^{\eta-s-1}| < 2^{\eta-s-1}$ holds since $0 < p_l < 2^{\eta-s}$. Since $q < p < 2\tilde{p}$, we can get

$$|q - \tilde{q}| = \frac{q \cdot 2|p_l - 2^{\eta-s-1}|}{2\tilde{p}} < 2|p_l - 2^{\eta-s-1}| < 2 \cdot 2^{\eta-s-1} = 2^{\eta-s}. \quad (7)$$

Let \tilde{q}_m be the s MSBs of $[\tilde{q}]$, that is

$$\tilde{q}_m = \left\lfloor \frac{[\tilde{q}]}{2^{\eta-s}} \right\rfloor = \left\lfloor \left\lfloor \frac{N}{p_m \cdot 2^{\eta-s} + 2^{\eta-s-1}} \right\rfloor / 2^{\eta-s} \right\rfloor.$$

From (7), we can get $|q_m - \tilde{q}_m| \leq 1$ (otherwise, $|q_m - \tilde{q}_m| \geq 2$, and so $|q - \tilde{q}| > 2^{\eta-s}$, which leads to a contradiction), and then (5) is obviously true.

Similarly, p_m can also be computed when q_m is known. Let $\tilde{q} = q_m \cdot 2^{\eta-s} + 2^{\eta-s-1}$ and $\tilde{p} = N/\tilde{q}$. Then

$$|p - \tilde{p}| = \left| p - \frac{N}{\tilde{q}} \right| = \frac{p|q - \tilde{q}|}{\tilde{q}} = \frac{p|q_l - 2^{\eta-s-1}|}{\tilde{q}}.$$

The inequality $|q_l - 2^{\eta-s-1}| < 2^{\eta-s-1}$ holds since $0 < q_l < 2^{\eta-s}$. We note that p, q, \tilde{q} are all n -bit numbers, and so it is clear that $p < 2\tilde{q}$. Therefore, we can get

$$|p - \tilde{p}| = \frac{p \cdot 2|q_l - 2^{\eta-s-1}|}{2\tilde{q}} < 2|q_l - 2^{\eta-s-1}| < 2^{\eta-s},$$

which implies that (6) is true with a same discussion as above. This completes the proof. \square

Remark 3 From Lemma 4 we know, for p_m and q_m , no matter which one is known, the other one can be computed successfully by the same method. Therefore, in our experiments, p_m and q_m have the same status and effect.

In our new practical attack, we will take $\alpha = 0$ (or take $\alpha' = 0$ if we need) in Lemma 4. Although $\alpha = 0$ does not always hold in practice, it will not be an obstruction to our practical attack, since good approximations of p_m and q_m can also help to realize a successful attack thanks to the multivalued-continuous phenomena which will be introduced in Subsection 4.3.

When s MSBs of p is enumerated, Equation (1) then becomes to

$$ed - 1 = k(N + 1 - (p_m + q_m) \cdot 2^{\eta-s} - (p_l + q_l)).$$

Let $A' = N + 1 - (p_m + q_m) \cdot 2^{\eta-s}$, $y' = -(p_l + q_l)$, $x' = x = k$. Then we have

$$f'(x', y') = x'(A' + y') + 1 \equiv 0 \pmod{e}.$$

Using the substitution $u' = x'y' + 1$, the equation above changes to

$$f'(u', x') = u' + A'x' \equiv 0 \pmod{e}.$$

Let $2^s = N^\xi$. Then the bounds of the new variables become to

$$x' < X' = N^\delta, \quad y' < Y' = N^{\frac{1}{2}-\xi},$$

$$e < N, \quad u' < U' = N^{\delta+\frac{1}{2}-\xi}.$$

A result similar to (4) can be obtained, i.e.,

$$\delta < \frac{-(\dim L)^2 \cdot \frac{\log_2 1.02}{2\eta} + \left(m - \frac{\log_2 \dim L}{4\eta}\right) \cdot \dim L - \left(\frac{1}{2} - \xi\right)(s_Y + s_U) - s_e}{s_X + s_U}. \quad (8)$$

Table 2 Partial numerical corresponding relation between δ and s with $m = 7$, $t = 3$, $2\eta = 1024$

s	$\xi(s/1024)$	δ
24	24/1024	0.2899
25	25/1024	0.2906
26	26/1024	0.2913
27	27/1024	0.2920
28	28/1024	0.2927
29	29/1024	0.2935
30	30/1024	0.2942

Table 3 Partial numerical corresponding relation between δ and s with $m = 12$, $t = 5$, $2\eta = 1024$

s	$\xi(s/1024)$	δ
16	16/1024	0.2910
17	17/1024	0.2917
18	18/1024	0.2924
19	19/1024	0.2931
20	20/1024	0.2938
21	21/1024	0.2946
22	22/1024	0.2953

Similar to Sarkar in [25], we hence adopt the discrete strategy to show the relationship numerically instead. Partial numerical corresponding relations between δ and s are given in Tables 2 and 3.

As can be seen from Tables 2 and 3, for a 1024-bit-modulus RSA, when $m = 7$ and $t = 3$, at least 27 MSBs of p need to be exhausted to raise the upper bound of δ to 0.292. Similarly, when $m = 12$ and $t = 5$, the necessary number of exhausted MSBs is 18 to achieve the same bound with a 1024-bit N . This is well consistent with our experiments. In details, when $m = 7$ and $t = 3$, the success rate with a 26-MSB exhaustion is 13%, while in sharp contrast the rate with 27-MSB exhaustion is 87%; when $m = 12$ and $t = 5$, the success rates of 17-MSB exhaustion and 18-MSB exhaustion are 50% and 94%. For more details, see Section 5.

Remark 4 Tables 2 and 3 will play an important role in our new practical attack which is introduced in Subsection 4.3. Since the necessary numbers of exhausted MSBs in Tables 2 and 3 are given by our estimations and well consistent with our experiments, they may be nice instructors for us to choose suitable parameters (namely m, t, s) to implement our attack.

Similar results can be seen in [22] and [25] and we display part of them below in Table 4. We note that the necessary numbers of exhausted MSBs in [22] are given by estimations while those in [25] are given by experiments. Moreover, the maximum number, the minimum number and the mean number of the necessary exhausted MSBs were provided in [25] by large amounts of experiments. From Table 4 we can see, 20-MSB exhaustion is needed to attain the bound $\delta \leq 0.291$ in [22] while the mean number of necessary exhausted MSBs to attain the bound $\delta \leq 0.290$ is 23.8 in [25]; our estimations show that 24-MSB exhaustion is needed to attain the bound

$\delta \leq 0.2903$, roughly the same as [25]. This implies that our estimation is reasonable and more consistent with implementations than [22].

Table 4 Partial corresponding relation between δ and s given in [22], [25] and ours with $m = 7, t = 3$

Suk 2003 [22]			Sarkar 2008 [25]			Ours		
N (bit)	s (bit)	δ	N (bit)	mean s (bit)	δ	N (bit)	s (bit)	δ
1000	10	0.285	1000	9	0.280	1000	20	0.2874
1000	20	0.291	1000	16.4	0.285	1000	21	0.2882
1000	30	0.298	1000	23.8	0.290	1000	22	0.2889
1000	40	0.304	1000	31.8	0.295	1000	23	0.2896
1000	50	0.308	1000	38.6	0.300	1000	24	0.2903

4.3 A new practical attack based on the binary search

When implementing our practical small private exponent attack on RSA, some interesting and nontrivial phenomena appear: (I) besides p_m and q_m , the real values of the MSBs of p and q , a value close to p_m or q_m can also help to attack RSA successfully in the exhaustion process; (II) these additional exhaustive values appear continuously around p_m and q_m (the word “continuously” here means continuous integral point in the interval); (III) the closer p and q are, the more additional exhaustive values there will be. We call the above phenomena the “multivalued-continuous phenomena”. The multivalued-continuous phenomena immediately inspire us a new attack based on the binary search. That is, to complete the attack effectively, we can try the binary search for the MSBs of p in the exhausted space. The binary search means that we need neither to go through the exhausted space from the smallest value to the largest ineffectively to find the real values of p_m and q_m , nor to know the exact amount of these values close to p_m and q_m who can help to attack RSA successfully. What we should do is just to efficiently find one of such values. Our experiments have verified the correctness and effectiveness of our algorithm based on the binary search.

4.3.1 The multivalued-continuous phenomena

At first we give a necessary lemma, which will be helpful to prove the multivalued-continuous phenomena.

Lemma 5 Let

$$h(x) = x + \left\lfloor \left\lfloor \frac{N}{bx + b/2} \right\rfloor / b \right\rfloor - c,$$

where x is a non-negative integer and N, b, c are positive integers. Then

$$h(x) \leq h(x-1) \quad \text{if } 1 \leq x \leq \left\lfloor \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rfloor; \quad \text{and}$$

$$h(x) \geq h(x-1) \quad \text{if } x \geq \left\lceil \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rceil.$$

Proof If $1 \leq x \leq \left\lceil \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rceil$, then $N \geq b^2(x+1/2)(x-1/2)$, and so

$$\frac{N}{b(x-1)+b/2} - \frac{N}{bx+b/2} = \frac{Nb}{(bx+b/2)(b(x-1)+b/2)} = \frac{Nb}{b^2(x+1/2)(x-1/2)} \geq b.$$

We note that $\lfloor u \rfloor - \lfloor v \rfloor \geq w$ naturally holds for any positive real numbers u, v and positive integer w with $u - v \geq w$. Therefore, we have

$$\left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor - \left\lfloor \frac{N}{bx+b/2} \right\rfloor \geq b,$$

and hence

$$\left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor / b - \left\lfloor \frac{N}{bx+b/2} \right\rfloor / b \geq 1,$$

which implies that

$$\left\lfloor \left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor / b \right\rfloor - \left\lfloor \left\lfloor \frac{N}{bx+b/2} \right\rfloor / b \right\rfloor \geq 1.$$

Note that

$$\begin{aligned} h(x-1) - h(x) &= x-1 + \left\lfloor \left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor / b \right\rfloor - x - \left\lfloor \left\lfloor \frac{N}{bx+b/2} \right\rfloor / b \right\rfloor \\ &= \left\lfloor \left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor / b \right\rfloor - \left\lfloor \left\lfloor \frac{N}{bx+b/2} \right\rfloor / b \right\rfloor - 1. \end{aligned}$$

Therefore, $h(x) \leq h(x-1)$ holds if $1 \leq x \leq \left\lceil \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rceil$.

If $x \geq \left\lceil \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rceil$, then $N \leq b^2(x+1/2)(x-1/2)$, and so

$$\frac{N}{b(x-1)+b/2} - \frac{N}{bx+b/2} = \frac{Nb}{b^2(x+1/2)(x-1/2)} \leq b.$$

With a similar discussion as above, we can get $h(x) \geq h(x-1)$. This completes the proof. \square

Remark 5 It is not difficult to see that, for $x \geq 1$,

$$h(x) - h(x-1) = \left\lfloor \left\lfloor \frac{N}{bx+b/2} \right\rfloor / b \right\rfloor - \left\lfloor \left\lfloor \frac{N}{b(x-1)+b/2} \right\rfloor / b \right\rfloor + 1 \leq 1$$

since $\frac{N}{bx+b/2} < \frac{N}{b(x-1)+b/2}$. Then combining with Lemma 5 we can conclude that: if

$x \geq \left\lceil \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rceil$, then $0 \leq h(x) - h(x-1) \leq 1$.

Let us consider the case where p_m is known. Let $A' = N + 1 - (p_m + q_m) \cdot 2^{\eta-s}$. Then

$$ed - 1 = x'(A' + y') \tag{9}$$

obviously holds if $y' = -(p_l + q_l)$, $x' = k$. Without loss of generality, **we assume $s > 2$ and the value of s we choose will make sure that d can be correctly recovered based on Equation (9).**

Let \tilde{p}_m be a guess value of p_m . Since the MSB of p is 1, we can assume that $2^{s-1} \leq \tilde{p}_m < 2^s$. Then by Lemma 4,

$$\tilde{q}_m = \left\lfloor \left\lfloor \frac{N}{\tilde{p}_m \cdot 2^{\eta-s} + 2^{\eta-s-1}} \right\rfloor / 2^{\eta-s} \right\rfloor \tag{10}$$

is a good approximation of q_m . Let

$$\Delta(x) = x + \left\lfloor \left\lfloor \frac{N}{bx + b/2} \right\rfloor / b \right\rfloor - (p_m + q_m), \quad x \in [2^{s-1}, 2^s), \quad (11)$$

where $b = 2^{\eta-s}$. It is clear that $\Delta(\tilde{p}_m) = (\tilde{p}_m + \tilde{q}_m) - (p_m + q_m)$. Let

$$\tilde{p}_l = p - \tilde{p}_m \cdot 2^{\eta-s}, \tilde{q}_l = q - \tilde{q}_m \cdot 2^{\eta-s} \text{ and } A'' = N + 1 - (\tilde{p}_m + \tilde{q}_m) \cdot 2^{\eta-s}.$$

Then

$$ed - 1 = x''(A'' + y'') \quad (12)$$

obviously holds if $y'' = -(\tilde{p}_l + \tilde{q}_l)$, $x'' = k$.

In our experiments, we find that there are some other guess values except p_m and q_m that can help to correctly recover d . The reason why these guess values exist may come from two aspects. The first one is that in most cases there exist some additional \tilde{p}_m besides p_m and q_m such that $\Delta(\tilde{p}_m) = 0$ (i.e., $\tilde{p}_m + \tilde{q}_m = p_m + q_m$), which implies that Equation (12) is the same as Equation (9). In this case d can be correctly recovered based on Equation (12). The second one is that the bound in Condition (1) of Lemma 2 is not so tight. In other words, in practical cases, when the values of some variables slightly exceed their presupposed upper bounds and other conditions remain unchanged, $g(\tilde{x}_1, \dots, \tilde{x}_r) = 0$ may still hold over \mathbb{Z} . This means that, in our experiments, even if $|\Delta(\tilde{p}_m)|$ is slightly larger than 0, the private exponent d may be correctly recovered based on Equation (12), though the value of $|-(\tilde{p}_l + \tilde{q}_l)|$ may exceed its presupposed upper bound $2^{\eta-s} = N^{\frac{1}{2}-\xi}$.

From the discussion above, if d can be correctly recovered based on Equation (12), the practical small private exponent attack will be successful. For convenience, a guess value \tilde{p}_m that can help to realize a successful small private exponent attack on RSA is called a **helpful guess value**. A pair $(\tilde{p}_m, \tilde{q}_m)$ is called a **helpful guess pair** if \tilde{p}_m is a helpful guess value. Such a set which consists of all the helpful guess values is called the **Helpful Guess Set**, denoted by Ω . Note that the parameter s we choose will guarantee that d can be correctly recovered based on Equation (9). Therefore, we have $\{p_m, q_m\} \subseteq \Omega$.

In order to present a better understanding of the ‘‘continuous’’ property, we need a heuristic assumption as follows.

Assumption 2 There exist two largest non-negative integers ρ_1 and ρ_2 such that: if $-\rho_1 \leq \Delta(\tilde{p}_m) \leq \rho_2$, then the private exponent d can be correctly recovered based on Equation (12) by the CopperSmith method with the help of the LLL algorithm and resultants computation.

Our experiments imply that ρ_1 and ρ_2 are small integers, which are closely related to specific experimental instances (i.e., N, e, δ), the dimension of the lattice constructed with the CopperSmith method and the ability of the LLL algorithm.

Now we give a strict statement of the ‘‘multivalued-continuous’’ phenomena, which is stated as follows.

Theorem 1 Let Ω denote the Helpful Guess Set. Let $\Delta(x)$ be defined as (11) and ρ_1, ρ_2 be the same as **Assumption 2**. Let $p_{min} \in [2^{s-1}, 2^s) \cap \mathbb{Z}$ satisfying that $\Delta(p_{min}) = \min\{\Delta(x) \mid x \in [2^{s-1}, 2^s) \cap \mathbb{Z}\}$.

(1) If $\Delta(p_{min}) \geq -\rho_1$, then there exist two integers $a_1 \leq a_2$ such that $\{a_1, a_1 + 1, \dots, a_2\} \subseteq \Omega$, where a_1, a_2 can be uniquely determined by ρ_2 satisfying that

$$\Delta(a_1) \leq \rho_2 < \Delta(a_1 - 1), \quad \Delta(a_2) = \rho_2 < \Delta(a_2 + 1).$$

(II) If $\Delta(p_{min}) < -\rho_1$, then there exist four integers $b_1 \leq b_2 < b_3 \leq b_4$ such that $\{b_1, b_1 + 1, \dots, b_2\} \cup \{b_3, b_3 + 1, \dots, b_4\} \subseteq \Omega$, where b_1, b_2, b_3, b_4 can be uniquely determined by ρ_1, ρ_2 satisfying that

$$\begin{aligned} \Delta(b_1) &\leq \rho_2 < \Delta(b_1 - 1), \quad \Delta(b_4) = \rho_2 < \Delta(b_4 + 1), \\ \Delta(b_2 + 1) &< -\rho_1 \leq \Delta(b_2), \quad \Delta(b_3 - 1) < \Delta(b_3) = -\rho_1. \end{aligned}$$

Proof We note first that by Remark 5, for an arbitrary integer ρ between $\Delta\left(\left\lceil\sqrt{\frac{N}{b^2} + \frac{1}{4}}\right\rceil\right)$ and $\Delta(2^s - 1)$, there exists at least one guess value \tilde{p}_{m_ρ} such that $\Delta(\tilde{p}_{m_\rho}) = \rho$.

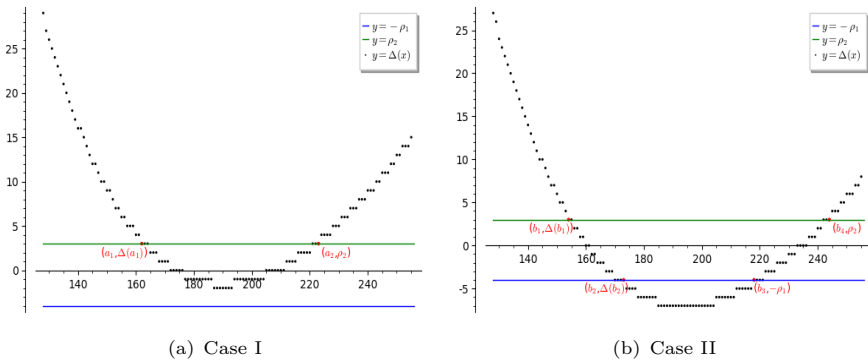


Fig. 1 A simple visualization of the two cases in the proof of Theorem 1

Case I: $\Delta(p_{min}) \geq -\rho_1$.

Let p_m, p_l, q_m, q_l be the same meanings as above and $b = 2^{\eta-s}$. Note that $p_m \geq q_m$. Since

$$\frac{N}{b^2} + \frac{1}{4} > \frac{N}{b^2} = \frac{(p_m b + p_l)(q_m b + q_l)}{b^2} \geq \frac{b^2 q_m^2}{b^2} = q_m^2,$$

it can be seen that $q_m \leq \left\lceil\sqrt{\frac{N}{b^2} + \frac{1}{4}}\right\rceil$. Note that $-\rho_1 \leq \Delta(q_m) \leq \rho_2$ since the value of s we choose will guarantee the success of our attack. Therefore, according to Lemma 5, if $\Delta(2^{s-1}) > \rho_2$, then there must exist an integer $a_1 \leq \left\lceil\sqrt{\frac{N}{b^2} + \frac{1}{4}}\right\rceil$ such that

$$\Delta(a_1) \leq \rho_2 < \Delta(a_1 - 1). \tag{13}$$

Meanwhile, according to Lemma 5 and the discussion at the beginning of this proof, if $\Delta(2^s - 1) > \rho_2$, there must exist an integer $a_2 \geq \left\lceil\sqrt{\frac{N}{b^2} + \frac{1}{4}}\right\rceil$ such that $\Delta(a_2) = \rho_2 < \Delta(a_2 + 1)$. For $\beta \in [a_1, a_2] \cap \mathbb{Z}$, we have

$$-\rho_1 \leq \Delta(p_{min}) \leq \Delta(\beta) \leq \max\{\Delta(a_1), \Delta(a_2)\} = \rho_2.$$

Therefore, according to **Assumption 2**, for any $\beta \in [a_1, a_2] \cap \mathbb{Z}$, the private exponent d can be correctly recovered based on Equation (12).

If $\Delta(2^{s-1}) \leq \rho_2$ or $\Delta(2^s - 1) \leq \rho_2$, we can take $a_1 = 2^{s-1}$ or $a_2 = 2^s - 1$ and get the same conclusion. A simple visualization of this case can be seen in Case I of Figure 1.

To finish the proof of Case I, it suffices to show the uniqueness of a_1 and a_2 . Suppose there exists another integer $a'_1 \leq \left\lfloor \sqrt{\frac{N}{b^2} + \frac{1}{4}} \right\rfloor$ such that

$$\Delta(a'_1) \leq \rho_2 < \Delta(a'_1 - 1). \quad (14)$$

If $a'_1 \geq a_1 + 1$, then it follows from Lemma 5 and Equations (13) and (14) that $\Delta(a'_1 - 1) \leq \Delta(a_1) \leq \rho_2 < \Delta(a'_1 - 1)$, a contradiction; if $a'_1 \leq a_1 - 1$, we can get that $\Delta(a'_1) \geq \Delta(a_1 - 1) > \rho_2 \geq \Delta(a'_1)$, still a contradiction. As a result, it must hold that $a'_1 = a_1$. The uniqueness of a_2 can be proved similarly.

Case II: $\Delta(p_{min}) < -\rho_1$.

If $\Delta(2^{s-1}) > \rho_2$ and $\Delta(2^s - 1) > \rho_2$, with a similar discussion as **Case I**, we can find $b_1 \leq b_2 < p_{min} < b_3 \leq b_4$ such that

$$\Delta(b_1) \leq \rho_2 < \Delta(b_1 - 1), \quad \Delta(b_4) = \rho_2 < \Delta(b_4 + 1),$$

$$\Delta(b_2 + 1) < -\rho_1 \leq \Delta(b_2), \quad \Delta(b_3 - 1) < \Delta(b_3) = -\rho_1.$$

For any $\beta \in \{[b_1, b_2] \cap \mathbb{Z}\} \cup \{[b_3, b_4] \cap \mathbb{Z}\}$, according to Lemma 5, we have

$$-\rho_1 = \min\{\Delta(b_2), \Delta(b_3)\} \leq \Delta(\beta) \leq \max\{\Delta(b_1), \Delta(b_4)\} = \rho_2.$$

If $\Delta(2^{s-1}) \leq \rho_2$ or $\Delta(2^s - 1) \leq \rho_2$, we can take $b_1 = 2^{s-1}$ or $b_4 = 2^s - 1$ and get the same conclusion.

A simple visualization of this case can be seen in Case II of Figure 1.

Similarly, the uniqueness of b_1, b_2, b_3, b_4 can be proved as **Case I**. This completes the proof of Theorem 1. \square

Table 5 Experimental presentation of the multivalued-continuous phenomena for $\delta \leq 0.292$

m	t	s	Amount of helpful guess values	Helpful guess values	Range of $\Delta(\bar{p}_m)$	Values of p, q, d
7	3	27	38	77936256~77936271 108904916~108904937	3~2 -2~3	$p=108791611503235673140185533091$ 50884154374894967508848133612045 43365870112083433518208166752126 34513230233723986376271419152033
7	3	28	55	155872521~155872543 217809833~217809864	4~4 -4~4	80518373404428155128956096567 $q=778551777306676461068475502186$ 11135114590047765139984700179502 00302423785621944272320203721259
10	4	21	43	1217740~1217757 1701633~1701657	4~2 -2~4	52886515236996817319552121527522 5005359166335680284144577427
12	5	18	36	152207~152221 212700~212720	4~1 -1~4	$d=822132486821443838429019508289$ 20810532559690610604683764553272 9448691328114281177912705023
10	4	21	3851	1060371~1064222	5~1~5	$p=679486791883996600331778827685$ 93480036059591634701532395565550 86565666723761153014913169297896 63406996117945403134009928504337
10	4	22	5819	2121683~2127502	3~0~3	6680194594851361518965438191 $q=678837169337166310745226733340$ 50992083124031396996898938020837 71485358806632262195329165541096
12	5	18	1276	132150~133425	4~1~4	64796783382129387505419864052820 9010624705809648621661183571
12	5	19	2093	264529~266621	4~0~4	$d=688450019926775649603539864984$ 76584008893082985088950620216788 3636867990095285712393863167

In order to give a more intuitive presentation of the phenomena, we provide partial specific experimental data in Table 5. As shown in Table 5, when $|\Delta(\tilde{p}_m)|$ is slight greater than 0, such \tilde{p}_m may still be a helpful guess value. For the first specific (p, q, d) , the set Ω contains two continuous intervals over \mathbb{Z} . For example, when $(m, t, s) = (7, 3, 27)$, we find 38 helpful guess values which appear as two parts around $p_m = 108904923$ and $q_m = 77936267$ respectively. For the second specific (p, q, d) , the set Ω contains one continuous interval over \mathbb{Z} . For example, when $(m, t, s) = (12, 5, 19)$, we find 2093 helpful guess values which appear as a continuous integer interval including $p_m = 265700$ and $q_m = 265446$.

Another phenomenon should be mentioned that there are much more helpful guess values when p is close to q in our experiments (one of the detailed comparison can be seen in Table 5, noting that the first $|p - q|$ is much greater than the second one). The “ p close to q ” contributes to the efficiency of our experiments pretty well. It must be mentioned that the use of “ p close to q ” in our attack is quite different from the attacks with small prime difference, in which the upper bound of $|p - q|$ should be known before implementing the attacks. While in our attack, we do not need to know how close between p and q in advance.

4.3.2 A new practical attack based on the binary search

Based on the multivalued-continuous phenomena, we propose a new practical small private exponent attack on RSA based on the binary search. The complete and detailed algorithm is displayed as follows.

Algorithm 1: New practical attack based on the binary search

Input: $N, e, \delta; m, t, s$

Output: p, q, d

0 $\eta \leftarrow \lfloor \log_2 p \rfloor$

1 **for** $j := 1, \dots, s - 1$

2 **for** $i := 1, \dots, 2^{j-1}$

3 $\tilde{p}_m \leftarrow 2^{s-1} + (2i - 1) \cdot 2^{s-1-j};$

4 $\tilde{q}_m \leftarrow \left\lfloor \left\lfloor \frac{N}{\tilde{p}_m \cdot 2^{\eta-s} + 2^{\eta-s-1}} \right\rfloor / 2^{\eta-s} \right\rfloor;$

5 $A \leftarrow N + 1 - (\tilde{p}_m + \tilde{q}_m) \cdot 2^{\eta-s};$

6 $Y \leftarrow \left\lfloor \sqrt{N}/2^s \right\rfloor, X \leftarrow N^\delta, U \leftarrow \left\lfloor \sqrt{N}/2^s \right\rfloor \cdot N^\delta;$

7 Run HM2010 attack with bounds in step 6;

8 **if** d is correctly found **then**

9 **return** $d, p, q;$

10 **end if**

11 **end for**

12 **end for**

We note that, the efficiency of Algorithm 1 relies on its input parameters m, t, s . Then a question may come up immediately that how should we choose appropriate m, t and s in order to attack the RSA cryptosystem efficiently.

According to our experiments and Miller’s discussion in [14], BKZ reduction algorithm doesn’t perform well in the HM2010 attack. Therefore, we adopt LLL algorithm in SageMath to carry out the lattice basis reduction in our attack.

It is not difficult to see that the total running time of our attack mainly comes from two parts: the search of a helpful guess value and the lattice basis reduction. In the following of this subsection, we first focus on the cost of LLL algorithm, and

then discuss how to find appropriate m , t and s based on some theoretical derivation and experimental results.

The time complexity of LLL algorithm and its variants is displayed below (the vector length, the number of basis vectors, the size of vector norm are denoted by n, d, B respectively).

LLL(1982)[18]	LLL(Schnorr 1988)[19]	L^2 (Nguyen 2005)[20]
$O(d^5 n B^3)$	$O(d^3 n (d + B)^2 B)$	$O(d^4 n (d + B) B)$

Making a specific analysis on the lattice we constructed, we get

$$n = d = \dim L' = \frac{(m+1)(m+2)}{2} + \frac{\tau m^2 + (2\tau - 1)m}{2}. \quad (15)$$

The size of the vector norm B is given by the following Proposition 1.

Proposition 1 With the notations as above, the approximate size of the largest vector norm B in our new lattice is $(m + \delta m + (\frac{1}{2} - \xi)\tau m) \log N$, where $2^s = N^\xi$.

Proof. See Appendix B.

It is not difficult to see that, the approximation size of B in Proposition 1 changes very little with s , since s is at most several dozens and the size of N is more than 1000 bits in our experiments. Note that the optimal t is determined and so is the parameter τ since $\tau = t/m$ for a fixed m . Therefore, based on Equation (15) and Proposition 1, the dimension remains unchanged and vector norm B is almost unchanged for fixed m . Above all, no matter whether MSBs guess strategy is added to HM2010 attack or not, the running time of the LLL algorithm is nearly unchanged, which is consistent with the results of our experiments.

Now we can try to determine the parameters m, t, s . Several groups of experiments are implemented and one of them is displayed in Table 6. The set Ω consists of two consecutive parts and the amount of helpful guess values is the sum of the two parts. For the same m and t , as s increases by 1, the amount of helpful guess values is less than twice as the original ones (all our experiments have verified the conclusion). This result indicates that the guess bit should try as few as possible. Meanwhile, after a simple computation we can conclude that an appropriately large m will reduce the total running time of our new attack. As can be seen from Table 1, when m increases continuously the improvement effect of upper bound of δ is less and less significant. Therefore, we would like to choose a medium m for our attack.

From the discussion above, to get a practical bound $\delta \leq 0.292$ for 1024-bit-modulus RSA, the recommended parameters we provide are $m = 12, t = 5, s = 18$. The reason why we don't recommend $s = 17$ is that the success rate cannot be guaranteed both from the estimation in Table 3 and the experimental results in Table 8. These parameters may not be the optimal ones, but they can help to implement our new attack very well to recover the private exponent for RSA with a 1024-bit modulus. With the same discussion, for 2048-bit-modulus RSA, the recommended parameters we provide are $m = 12, t = 5, s = 19$ to attain the bound $\delta \leq 0.287$.

5 Experiments

All our experiments are implemented in SageMath 9.1 on our PC with Intel(R) Xeon(R) W-2255 CPU (3.70GHz, 160GB RAM Windows 10). The

Table 6 Helpful guess value and running time with different m, t, s for a specific p, q, d

m	t	s	Amount of helpful guess values	Running time (second)	Values of p, q, d
7	3	27	$88+83=171$	22	$p =$ 10919978044315432962287449 87495252293298287669810026 59912781166668189878340060 83751730218810152667600528 00195399170377886135507507 2818459763054393669447049 $q =$ 10295169880070613266125506 25471632556952314949178730 85284380593751570671860226 57231423722833261496785534 27488924052692488680353874 1732160096107231687821563 $d =$ 89299647071100914355096956 62873009629732090303014427 02686428014854729474633080 697710968831
7	3	28	$123+116=239$	22	
7	3	29	$175+165=340$	22	
7	3	30	$245+231=476$	22	
10	4	21	$122+115=237$	405	
10	4	22	$157+148=305$	405	
10	4	23	$209+197=406$	405	
12	5	17	$70+66=136$	1660	
12	5	18	$84+82=166$	1660	
12	5	19	$122+115=237$	1660	

codes of experiments which are shown in Tables 7 to 10 can be seen in <https://pastebin.com/zpUkrfDh>.

We carry out some experiments to research the practical bound of δ based on HM2010 attack at first. Our experimental results on the upper bound of δ and some comparison with [11, 12, 14, 21] are provided in Table 7.

Table 7 Our experimental results on the upper bound of δ and some comparison with [11, 12, 14, 21]

	$\log_2 N$	δ	Parameters	Dim	Running time	Success%
[11]	1000	0.265	$(m, t) = (5, 3)$	39	45 minutes	/
[12]	1000	0.278	$(m, t) = (11, 5)$	72	6 days	/
[21]	1024	0.277	$(m, t) = (7, 3)$	45	2.5 hours	/
[14]	1000	0.279	$(m, t, \sigma, \tau) = (10, 4, 2, -3)$	92	/	62%
[14]	1000	0.280	$(m, t, \sigma, \tau) = (10, 4, 2, -3)$	92	/	1%
Ours	1024	0.284	$(m, t) = (21, 9)$	304	89 hours	63%
Ours	1024	0.285	$(m, t) = (25, 10)$	429	16.4 days	60%

It should be made clear that the upper bound of δ we get in Table 7 is just based on HM2010 attack. We implement 100 experiments for the bound 0.284. And for the bound 0.285, 5 experiments have been done since the running time is too long. From the experiments, we get a nice bound in an acceptable running time with a reasonable success rate. Moreover, for a no-less-than-1000-bit modulus N , this bound is the best practical one for this kind of attacks as far as we know.

In order to investigate the necessary number of exhausted MSBs that can guarantee the success of the attack, we implement some experiments to verify the estimations for $\delta = 0.292$ in Tables 2 and 3, which is displayed in Table 8. As can be seen in Table 8, when $m = 7$, we should choose $s = 27$ considering the comparison of success rates; similarly, when $m = 12$, s should be chosen to 18 for the same reason.

Table 8 Experimental comparison of success rates with $\delta = 0.292$, $2\eta = 1024$ when $m = 7$ and $m = 12$

m	t	s	number of experiments	number of successes	success rate
7	3	26	100	13	13%
7	3	27	100	87	87%
12	5	17	50	25	50%
12	5	18	50	47	94%

Based on Algorithm 1, we implement our new practical attack. The results are shown in Table 9 below.

Table 9 Experimental results of our new practical attack on RSA with $d \leq N^{0.292}$ for 1024-bit moduli

	Parameters	Real MSBs (p_m, q_m)	Helpful guess pair (\tilde{p}_m, \tilde{q}_m)	Total running time
Exp. 1	$m = 12$ $t = 5$ $s = 18$	(132850, 132723)	(133120, 132454)	2.3 hours
Exp. 2		(194574, 183696)	(194560, 183709)	4.1 hours
Exp. 3		(255247, 152844)	(255248, 152843)	21.6 days

All the three experiments above successfully recover the private exponent of 1024-bit-modulus RSA with $d \leq N^{0.292}$ within a month, which is certainly an acceptable running time for a practical attack on a cryptosystem. The validity of our algorithm and the rationality of the recommended parameters are verified by the success of the experiments. As we can see from Table 9, the helpful guess value \tilde{p}_m who contributes to the recovery of the private exponent actually does not equal to p_m . It needs still long to reach the real value p_m by the binary search. The smaller $|p - q|$ is, the more helpful guess values there will be, thus leads to a shorter total running time. Detailed parameters of the three experiments are shown in Appendix C.

As can be seen in [25], to implement an experiment for $\delta \approx 0.285$ with a 1000-bit modulus, they need about a week with a cluster of 26 machines. While in our implementations for $\delta \approx 0.292$ with a 1024-bit modulus, we can successfully complete them in about 3 weeks with a single PC. Particularly, the running time may reach several hours when p is appropriately close to q (e.g., p shares just several MSBs with q in Exp. 1 in Table 9).

At last, we apply Algorithm 1 to a 2048-bit-modulus RSA. The improving effect of upper bound of δ is not so good as the 1024-bit one since the proportion $s/\log_2 N$ decreases for the same amount of guess bits. Moreover, the running time becomes

about triple longer since the data size B in LLL reduction is twice as the original. Three experiments are carried out for 2048-bit-modulus RSA, which are displayed in Table 10.

Table 10 Experimental results of our new practical attack on RSA with 2048-bit moduli

	Parameters	δ	Real MSBs (p_m, q_m)	Helpful guess pair (\tilde{p}_m, \tilde{q}_m)	Total running time
Exp. 4	$m = 12$ $t = 5$ $s = 19$	0.287	(363847, 306431)	(306432, 363846)	7.1 days
Exp. 5		0.287	(514229, 312777)	(514240, 312770)	35.8 days
Exp. 6		0.292	(7263440332, 7263440332)	(7263485952, 7263394712)	11.2 days

From Table 10 we can see that, for random primes p and q (such as p, q in Exp. 5), a majority of the 2048-bit-modulus RSA can be successfully attacked in about a month for $\delta \leq 0.287$; if p is slightly close to q (e.g., in Exp. 4, p, q share 3 MSBs), the efficiency will be better than random case; while p is quite close to q (e.g., in our setting in Exp. 6, p and q 50 MSBs), we can effectively recover the private exponent when $\delta \leq 0.292$. Detailed parameters can be seen in Appendix D.

6 Conclusion

In this paper, we focus on the practical small private exponent attack on RSA. After some detailed and relatively exact calculations of related parameters in the lattice determinant, we give a few precise estimations about the upper bound of solvable δ based the experimental LLL estimation by Nguyen et al.. With the instruction of our estimations, we implement the HM2010 attack and get a better bound of δ than the former results as far as we know. To improve the bound we can achieve by the HM2010 attack, we add a simple idea of the MSBs guess of p . Based on the nontrivial and inspiring multivalued-continuous phenomena, we propose a new attack based on the binary search and succeed to attack the 1024-bit-modulus RSA cryptosystem for $\delta \leq 0.292$. Additionally, a slightly weaker result $\delta \leq 0.287$ can be obtained after applying our new attack to 2048-bit-modulus RSA, which is also the best one among this kind of attacks as far as we know.

Furthermore, there are still some questions to be answered. Since the number of helpful guess values has close relation to the value of $|p - q|$ from the experiments, how should we exactly understand the relationship? How can we estimate the lower bound of the size of Ω ? As the effect of our new attack turns weak in a 2048-bit-modulus RSA attack, what else should we try to break through the 0.292 bound for a mainstream practical RSA? Although the gaps still exist, we believe our attack can provide some inspiration to search more practical attacks on RSA.

Acknowledgments. This research was supported by NSF of China under Grant No. 61872383.

References

- [1] Rivest R.L., Shamir A., Adleman L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM. 21(2),

- 120-126 (1978).
- [2] Wiener M.J.: Cryptanalysis of short RSA secret exponents. *IEEE Trans. Inf. Theory* 36(3), 553-558 (1990).
 - [3] Blömer J., May A.: A generalized wiener attack on RSA. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. *Lecture Notes in Computer Science*, vol. 2947, pp. 1-13. Springer, Heidelberg (2004).
 - [4] Bunder M., Tonien J.: A new attack on the RSA cryptosystem based on continued fractions. *Malaysian Journal of Mathematical Sciences*. 11 (S3), 45-57 (2017).
 - [5] Susilo W., Tonien J., Yang G.: The Wiener attack on RSA revisited: a quest for the exact bound. *Lecture Notes in Computer Science*, vol. 11547, pp. 381-398. Springer, Cham (2019).
 - [6] Boneh D., Durfee G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. *IEEE Trans. Inf. Theory* 46(4), 1339-1349 (2000).
 - [7] 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. *Lecture Notes in Computer Science*, vol. 6056, pp. 53-69. Springer, Heidelberg (2010)
 - [8] Herrmann M., May A.: Attacking power generators using unravelled linearization: when do we output too much? In: Matsui, M. (ed.) ASIACRYPT 2009. *Lecture Notes in Computer Science*, vol. 5912, pp. 487-504. Springer, Heidelberg (2009).
 - [9] Hinek M. J.: *Cryptanalysis of RSA and Its Variants*. Taylor and Francis group: CRC Press, 2009.
 - [10] Nitaj A., Ariffin M.R.K., Adenan N.N.H., et al.: Exponential increment of RSA attack range via lattice based cryptanalysis. *Multimedia Tools and Applications* 81, 36607-36622 (2022).
 - [11] Boneh D., Durfee G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$, In: Stern, J. (ed.) EUROCRYPT 1999. *Lecture Notes in Computer Science*, vol. 1592, pp. 1-11. Springer, Heidelberg (1999).
 - [12] Blömer J., May A.: Low secret exponent RSA revisited. In: Silverman, J.H. (ed.) CaLC 2001. *Lecture Notes in Computer Science*, vol. 2146, pp. 4-19. Springer, Heidelberg (2001).
 - [13] David Wong: [https://github.com /mimoo/RSA-and-LLL-attacks](https://github.com/mimoo/RSA-and-LLL-attacks).

- [14] Miller S.D., Narayanan B., Venkatesan R.: Coppersmith's lattices and "focus groups": An attack on small-exponent RSA, *Journal of Number Theory*. 222, 376-392 (2021).
- [15] Nguyen P., Stehlé D.: LLL on the average. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTSVII. *Lecture Notes in Computer Science*, vol. 4076, pp. 238-256. Springer, Heidelberg (2006).
- [16] Coppersmith, D.: Finding a small root of a univariate modular equation. In: U. M. Maurer (ed.) EUROCRYPT 1996. *Lecture Notes in Computer Science*, vol. 1070, pp. 155-165. Springer, Heidelberg (1996).
- [17] Howgrave-Graham N.: Finding small roots of univariate modular equations revisited. In: Darnell M.J. (ed.) *Cryptography and Coding 1997*. *Lecture Notes in Computer Science*, vol. 1355, pp. 131-142. Springer, Heidelberg (1997).
- [18] Lenstra A.K., Lenstra H.W., Lovsz L.: Factoring polynomials with rational coefficients. *Math. Ann.* 261(4), 515-534 (1982).
- [19] Schnorr C. P.: A more efficient algorithm for lattice basis reduction. *Journal of Algorithms*. 9(1), 47-62 (1988).
- [20] Nguyen P. Q., Stehlé D.: Floating-point LLL revisited. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. *Lecture Notes in Computer Science*, vol. 3494, pp. 215-233. Springer, Heidelberg (2005).
- [21] Durfee G.: Public key cryptanalysis using algebraic and lattice methods. Ph.D.thesis, Stanford University, Stanford (2002).
- [22] Suk A. H.: Cryptanalysis of RSA with lattice attacks. Ph.D.thesis, University of Illinois, Illinois (2003).
- [23] Rivest R. L., Shamir A.: Efficient factoring based on partial information. In: Pichler, F. (ed.) EUROCRYPT 1985. *Lecture Notes in Computer Science*, vol. 219, pp. 31-34. Springer, Heidelberg (1986).
- [24] Coppersmith D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptol.* 10(4), 233-260 (1997).
- [25] Sarkar S., Maitra S., and Sarkar S.: RSA cryptanalysis with increased bounds on the secret exponent using less lattice dimension. *IACR Cryptology ePrint Archive*, 315 (2008).
- [26] Sarkar S, Maitra S: Improved partial key exposure attacks on RSA by guessing a few bits of one of the prime factors. In: Lee, P.J., Cheon, J.H. (eds.) ICISC 2008. *Lecture Notes in Computer Science*, vol. 5461, pp.

- 37-51. Springer, Heidelberg (2009).
- [27] Liu C., Yang C.: Factoring RSA modulo N with high bits of p known revisited. 2009 IEEE International Symposium on IT in Medicine & Education. IEEE, 1, 495-500 (2009).
- [28] Lu Y., Zhang R., Lin D.: Factoring RSA modulus with known bits from both p and q : A Lattice Method. In: Lopez J., Huang X., Sandhu R. (eds) Network and System Security 2013. Lecture Notes in Computer Science, vol 7873, pp. 393-404. Springer, Heidelberg (2013).
- [29] Hashimoto Y: On small secret key attack against RSA with high bits known prime factor. Cryptology ePrint Archive, 2010.
- [30] Peng L., Hu L., Huang Z., et al.: Partial prime factor exposure attacks on RSA and its Takagi's variant. In: Lopez, J., Wu, Y. (eds.) ISPEC 2015. Lecture Notes in Computer Science, vol. 9065, pp. 96-108. Springer, Cham (2015).
- [31] Takayasu A., Kunihiro N.: A tool kit for partial key exposure attacks on RSA. In: Handschuh, H. (ed.) CT-RSA 2017. Lecture Notes in Computer Science, vol. 10159, pp. 58-73. Springer, Cham (2017).
- [32] Coppersmith D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 178-189. Springer, Heidelberg (1996).

Appendix A Proof of Lemma 3

Proof Take $\tau=m/t$, then we have

$$\begin{aligned}
 s_X &= \sum_{k=0}^m \sum_{i=0}^{m-k} i = \sum_{k=0}^m \frac{(m-k)(m-k+1)}{2} \\
 &= \frac{1}{2} \sum_{k=0}^m (m(m+1) - (2m+1)k + k^2) \\
 &= \frac{1}{2} \left(m(m+1)^2 - (2m+1) \frac{m(m+1)}{2} + \frac{m(m+1)(2m+1)}{6} \right) \\
 &= \frac{m(m+1)(m+2)}{6}, \\
 s_Y &= \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m j = \sum_{j=1}^{\tau m} j \left(m - \left\lfloor \frac{1}{\tau} \right\rfloor j + 1 \right) \\
 &= \sum_{j=1}^{\tau m} (m+1)j - \sum_{j=1}^{\tau m} \left\lfloor \frac{1}{\tau} \right\rfloor j^2
 \end{aligned}$$

$$\begin{aligned}
&= \frac{(m+1)\tau m(\tau m+1)}{2} - \left\lfloor \frac{1}{\tau} \right\rfloor \cdot \frac{\tau m(\tau m+1)(2\tau m+1)}{6} \\
&\approx \frac{(m+1)\tau m(\tau m+1)}{2} - \frac{1}{\tau} \cdot \frac{\tau m(\tau m+1)(2\tau m+1)}{6} \\
&= \frac{\tau^2 m^3 + 3\tau^2 m^2 + 3\tau m - m}{6}, \\
s_U &= \sum_{k=0}^m \sum_{i=0}^{m-k} k + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m k \\
&= \sum_{k=0}^m k(m-k+1) + \sum_{j=1}^{\tau m} \frac{(m + \lfloor \frac{1}{\tau} \rfloor j)(m - \lfloor \frac{1}{\tau} \rfloor j + 1)}{2} \\
&\approx \sum_{k=0}^m k(m-k+1) + \sum_{j=1}^{\tau m} \frac{(m + \frac{1}{\tau} j)(m - \frac{1}{\tau} j + 1)}{2} \\
&= \sum_{k=0}^m k(m+1) - \sum_{k=0}^m k^2 + \frac{1}{2} \sum_{j=1}^{\tau m} (m + \frac{1}{\tau} j)(m - \frac{1}{\tau} j + 1) \\
&= \sum_{k=0}^m k(m+1) - \sum_{k=0}^m k^2 + \frac{1}{2} \sum_{j=1}^{\tau m} \left(m(m+1) + \frac{1}{\tau} j - \frac{1}{\tau^2} j^2 \right) \\
&= \frac{m(m+1)^2}{2} - \frac{m(m+1)(2m+1)}{6} \\
&\quad + \frac{1}{2} \left(m(m+1)\tau m + \frac{\tau m(\tau m+1)}{2\tau} - \frac{\tau m(\tau m+1)(2\tau m+1)}{6\tau^2} \right) \\
&= \frac{m(m+1)(m+2)}{6} + \frac{(4\tau^2 m^3 + (9\tau^2 - 3\tau)m^2 + (3\tau - 1)m)}{12\tau} \\
&= \frac{(4\tau^2 + 2\tau)m^3 + (9\tau^2 + 3\tau)m^2 + (7\tau - 1)m}{12\tau}, \\
s_e &= \sum_{k=0}^m \sum_{i=0}^{m-k} (m-k) + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m (m-k) \\
&= \sum_{k=0}^m (m-k)(m-k+1) \\
&\quad + \sum_{j=1}^{\tau m} \left(m(m - \lfloor \frac{1}{\tau} \rfloor j + 1) - \frac{(m + \lfloor \frac{1}{\tau} \rfloor j)(m - \lfloor \frac{1}{\tau} \rfloor j + 1)}{2} \right) \\
&\approx \sum_{k=0}^m (m-k)(m-k+1) \\
&\quad + \sum_{j=1}^{\tau m} \left(m(m - \frac{1}{\tau} j + 1) - \frac{(m + \frac{1}{\tau} j)(m - \frac{1}{\tau} j + 1)}{2} \right) \\
&= \sum_{k=0}^m (m(m+1) - (2m+1)k + k^2) + \frac{1}{2} \sum_{j=1}^{\tau m} \left((m - \frac{1}{\tau} j)(m - \frac{1}{\tau} j + 1) \right)
\end{aligned}$$

$$\begin{aligned}
&= m(m+1)^2 - \frac{(2m+1)m(m+1)}{2} + \frac{m(m+1)(2m+1)}{6} \\
&\quad + \sum_{j=1}^{\tau m} \left(m(m+1) - (2m+1)\frac{1}{\tau}j + \frac{1}{\tau^2}j^2 \right) \\
&= \frac{m(m+1)(m+2)}{3} \\
&\quad + \frac{1}{2} \left(m(m+1)\tau m - \frac{(2m+1)\tau m(\tau m+1)}{2\tau} + \frac{\tau m(\tau m+1)(2\tau m+1)}{6\tau^2} \right) \\
&= \frac{m(m+1)(m+2)}{3} + \frac{2\tau^2 m^3 + (3\tau^2 - 3\tau)m^2 - (3\tau - 1)m}{12\tau}, \\
\dim L &= \sum_{k=0}^m \sum_{i=0}^{m-k} 1 + \sum_{j=1}^{\tau m} \sum_{k=\lfloor \frac{1}{\tau} \rfloor j}^m 1 \\
&= \sum_{k=0}^m (m-k+1) + \sum_{j=1}^{\tau m} \left(m - \left\lfloor \frac{1}{\tau} \right\rfloor j + 1 \right) \\
&\approx \sum_{k=0}^m (m-k+1) + \sum_{j=1}^{\tau m} \left(m - \frac{1}{\tau}j + 1 \right) \\
&= (m+1)^2 - \frac{m(m+1)}{2} + \tau m(m+1) - \frac{1}{\tau} \cdot \frac{\tau m(\tau m+1)}{2} \\
&= \frac{(m+1)(m+2)}{2} + \frac{\tau m^2 + (2\tau - 1)m}{2}.
\end{aligned}$$

The proof of Lemma 3 has completed. \square

Appendix B Proof of Proposition 1

Proof It must be pointed out that the largest size of the vector norm B can be perfectly approximated by the maximum component of all the vectors in the lattice basis, e.g., the maximum coefficient of the term in all the polynomials.

At first we compute B in the lattice of the HM2010 attack. Note that vectors are produced by x -shift polynomials and y -shift polynomials:

$$\begin{aligned}
\tilde{g}_{i,k}(u, x) &= x^i \tilde{f}^k e^{m-k}, \quad i = 0, \dots, m-k, \quad k = 0, \dots, m \\
\tilde{h}_{j,k}(u, x, y) &= y^j \tilde{f}^k e^{m-k}, \quad j = 1, \dots, \tau m, \quad k = \lfloor 1/\tau \rfloor j, \dots, m,
\end{aligned}$$

where $\tilde{f}(u, x) = A + ux$ and $t = \tau m$.

(I) The size of the maximum coefficient in x -shift polynomials

Note that

$$\tilde{g}_{i,k}(u, x) = x^i \tilde{f}^k e^{m-k} = \sum_{a=0}^k \binom{k}{a} x^i u^a A^{k-a} x^{k-a} e^{m-k}.$$

Terms in $\tilde{g}_{i,k}(u, x)$ are of the form $\binom{k}{a} u^a A^{k-a} x^{k-a+i} e^{m-k}$. The final coefficients of such terms (i.e., the values of vector components when constructing lattices) is $\binom{k}{a} U^a A^{k-a} X^{k-a+i} e^{m-k}$ with a size of

$$\left(a\left(\frac{1}{2} + \delta\right) + k - a + (k - a + i)\delta + m - k\right) \log N = \left(m - \frac{1}{2}a + (k + i)\delta\right) \log N.$$

The size of the maximum coefficient is $(m + m\delta) \log N$ according to the formula above since $0 \leq a \leq k < m, 0 \leq k + i \leq k + m - k = m$.

(II) The size of the maximum coefficient in y -shift polynomials

Note that Terms in $\tilde{h}_{j,k}(u, x, y)$ is of the form $\binom{k}{b} y^j u^b A^{k-b} x^{k-b} e^{m-k}$.

Case A:

When $j \geq k - b$,

$$\sum_{b=0}^k \binom{k}{b} y^j u^b A^{k-b} x^{k-b} e^{m-k} = \sum_{b=0}^k \binom{k}{b} y^{j-(k-b)} u^b A^{k-b} e^{m-k} (u-1)^{k-b}.$$

Therefore, the term which has the maximum coefficient is $\binom{k}{b} y^{j-(k-b)} A^{k-b} u^k e^{m-k}$ with the final coefficient $\binom{k}{b} Y^{j-(k-b)} A^{k-b} U^k e^{m-k}$. In this case the size of the maximum coefficient is

$$\left(\frac{1}{2}(j - (k - b)) + k\left(\frac{1}{2} + \delta\right) + k - b + m - k\right) \log N = \left(\frac{1}{2}j - \frac{1}{2}b + \delta k + m\right) \log N.$$

We get $(\frac{1}{2}j - \frac{1}{2}b + \delta k + m) \log N \leq (\frac{1}{2}j - \frac{1}{2}b + \delta j + \delta b + m)$ for $k \leq j + b$. Finally, the size of the maximum coefficient is $(m + \tau\delta m + \frac{1}{2}\tau m) \log N$ when $j = \tau m, b = 0$ as $0 \leq b \leq k \leq m, 1 \leq j \leq \tau m$.

Case B:

When $j < k - b$, after a similar discussion as **Case A**, we can get the size of the maximum coefficient in $y^j \tilde{f}^k e^{m-k}$ is

$$\left((b + j)\left(\frac{1}{2} + \delta\right) + k - b + (k - b - j)\delta + m - k\right) \log N = \left(\frac{1}{2}j - \frac{1}{2}b + \delta k + m\right) \log N.$$

Finally, since $j < k - b, 0 \leq b \leq k, 1 \leq j \leq \tau m, \lfloor 1/\tau \rfloor j \leq k \leq m$, the size of the maximum coefficient is $(m + \delta m + \frac{1}{2}\tau m) \log N$ when $j = \tau m, b = 0, k = m$.

According to the detailed discussion, we obtain that

$$\begin{aligned} B &= \max\{(m + m\delta) \log N, (m + \tau\delta m + \frac{1}{2}\tau m) \log N, (m + \delta m + \frac{1}{2}\tau m) \log N\} \\ &= (m + \delta m + \frac{1}{2}\tau m) \log N. \end{aligned}$$

At the second step we compute B in our lattice. Use the notations above, the upper bound of each variable becomes to

$$\begin{aligned} x' &< X' = N^\delta, \quad y' < Y' = N^{\frac{1}{2}-\xi}, \\ e &< N, \quad u' < U' = N^{\delta+\frac{1}{2}-\xi}, \end{aligned}$$

where $2^s = N^\xi$ and s is the amount of MSBs exhaustion. Note that the terms in the polynomials do not change while the bounds of variables differs comparing the lattice of HM2010 with ours. With a simple analysis as former, we can obtain the maximum sizes of coefficient in x' -shift polynomials and y' -shift polynomials are $(m + m\delta) \log N$ and $(m + \delta m + (\frac{1}{2} - \xi)\tau m) \log N$. Finally we obtain $B = (m + \delta m + (\frac{1}{2} - \xi)\tau m) \log N$.

Based on the discussion above, we can see that the largest size of vector norm B is $(m + \delta m + (\frac{1}{2} - \xi)\tau m) \log N$. The proof of Proposition 1 has completed. \square

Appendix C Detailed parameters of the three experiments in Table 9

Exp. 1:

$N =$

46126089040452448339448600417060313922101098426244293259787635
 87074530673491676174094741308570008991308086429133694253082502
 08322979383825756505663848479411358228528222527395327435101891
 24124500359010333442383692988340095271183825614638791911699269
 162046275028679426500348785157345976662456325437259705160061

$e =$

36614584641331081308456049556562616703353184435474954472543851
 64159874655507513240178939924241301243795285290427026969984191
 93417336370186256151741426122690631967234543279797441311494545
 48205774006091224643569311902728986446145416739772661245661572
 872595702072188094041649860081717657262961694486055770442903

$p =$

67948679188399660033177882768593480036059591634701532395565550
 86565666723761153014913169297896634069961179454031340099285043
 376680194594851361518965438191

$q =$

67883716933716631074522673334050992083124031396996898938020837
 71485358806632262195329165541096647967833821293875054198640528
 209010624705809648621661183571

$d =$

68845001992677564960353986498476584008893082985088950620216788
 3636867990095285712393863167

Exp. 2:

$N =$

93502450932903310633064573151907317024294867669134678632093362
 46297788099166895737544153127851852138932264560083819167132205
 42943047406167744180006421454219231575086254775401435086568101
 80893794178324182089403622394064939255883017451942933893424910
 850870147643227272288485474147840684850717753717586472451201

$e =$

66858620224726705843141951973966747720864781039484831550691634
 09294281451926768980157225811154880346715028924829626919171141
 60786364943066431341308061576266724936876531701099477141597025

75928828432166081285992511952702707653041998543208057095834381
069598542543859865005532560613469773387993762913083830903645

$p =$

99518569133681654608883617987619781762536839523918823141580161
9712383537617858184400864458688990504895522245453609041221097
121126837776453250821684510563

$q =$

93954778235710974203677377216985836253852371657508142242651124
33882347772961954728089804437932682899351880809635985985823721
278448590959326657930561582027

$d =$

84621421118866086279480677455817189905395785567701311624227231
1992184907517426187202723837

Exp. 3:

$N =$

10205793884912309428243538885795108687967400364358859068757105
27366676256258700648163154922583229485417321567269849917383364
64656067954130787989822823376442710598097310129836268241537114
71952829908327055972448207678376586518255364121854918727930884
3770512577006453837517293155340490761206708661611812914183899

$e =$

80900458040712014796201499026385729329804135026384835491013425
86092605732136485389209880857013543736344572515888256253718707
08436780537181756263571337006232687007715876604623603003318929
36401425760635712653711506424050436067133668344381418365690036
931579614717233156905341293247889945042931394909865508914479

$p =$

130550858861846841747727958575739547701679174283776488146464000
035126653179894990637272044524427541060375052834227051238884085
86029300473284150150163760489

$q =$

781748505822731655261191706621539039622764767903458858428781536
263460148861416123276603543978929241699568615263385404823218907
1292155782177922355115574691

$d =$

868126999952902548361525189983896174511773826207382896837234598
157112101757767726001029119

Appendix D Detailed parameters of the three experiments in Table 10

Exp. 4:

$N =$

13108249020538785310663414698582637064820950198561878706735851
 65703644339810427940041439226962830915168487281788869208017778
 64923887358082167988616026368541195282045124591109789444274263
 26718715032735164557685449265721895500116475985815536552066383
 98902635748400290357538362875036895327848802671465354828268316
 19459190428496888414853786863007208331552517424436288121969788
 57993634225342593766658498158262129054566502609859174903015093
 10543060200243513864877744474764130353698543557524799219837216
 81177688022836416789341375825625350075504640416012415003418082
 73695609938403611734357471580735947449865433515838599015767

$e =$

89451479500251299951266471597772406719039486388733814263161727
 77277693744087994022236345476762214316234081707614457014149993
 07086353862705412696231771650983658024823969641299772311875452
 35751367575578627668994418983506635558378045495835658481986197
 64498956959651824723201619704850371101473507734971701102683193
 24725402870657449338885605852347781947910706904405925408817023
 60111680067003156460400937994396681812731602263538839593687345
 62219589235850846108119781315380911546600335260644624325458166
 47435720465040339844832330681577201708958823066483795645154297
 8645874176685545024632447680732323591071139368778498577639

$p =$

12475716189463277471168125160458453539576717759904860818958950
 1201482534891974371957818466977821230125550070704404895576204
 52941396889634025921970794255590369593505603793752936553527919
 38379990938303607598997653011866480954848015397181895617491220
 3077504115908974119391542269233365957211095336385587786710643

$q =$

10507011238047985984905944714077857914018323916121653018787711
 19630266681204488009378026133638912480941505942176252858307437
 32991043217939651634566680101998984493904772645000773580396638
 66392564560306397986626626825696478790443273047686349310580634
 5028770283044102294594624281933734776943133629178895741769869

$d =$

16203833775447352356853531181274614215508930119487698497783433
 20417126835821821597640064942591829248499538916881397751938428
 70347081603575753080006255052855504063533452418351103

Exp. 5:

$N =$

18909672181904151395097968169920434800151774976081336450891174
 87637992213220042585977009330732380044765810427401976038590837
 11873143826745772602607516127893011661052190642253756693227076
 98926584779010796261564768643160271391197294787916627864241401
 91442506166372883954050983092065122469794049346568026032600851
 06520619119309685394418339932570134971764214265160652504821910
 54728868131847284573195025675911240577689925121207663441802463
 95424763276465590987069458679529387973710595569647138338589685
 41003463475729539029975985698319117707054049488848688491897152
 76445558134402328090544913847971380379882458726005541831877

$e =$

71750525858644602573827151612109334198329801833575974634891308
 81495398383619485101066743946892399989979056836606216684862725
 57211251358518687663563498732052487860671897005525731875587709
 48285298803158025787222714686015007936511780408515333161477308
 33938515599770436750069222320451660592526204534805678560379611
 66401313318683780361047153401836775380248541241640858508425424
 17567958496040977474953521700071041506080838448030690016792476
 72933113105225110610011794291511678512931585633775157710522063
 46208952124338836289923112469153753848820848393483936342199771
 0031387152470262859696222361705999483806613844394759888651

$p =$

17632053007251238779460491844609267953184637057432293238819260
 45182034840480286908458849795483938148101996899679104684164224
 74443986695404811633893785378170226481910000920070470403858119
 63853022804280585705368356890551045545683993527388328068472661
 0142603148675130345130093490471284531708421254883794726933731

$q =$

10724600348086231138978615424494121452147302572884539393558466
 49494608032097842525445414641585733688186933655325569963690234
 73297386324910499142225474129024782995490552263782465106021625
 38318333121616296839163140328592966430893795983414382893135148
 9815771244076937362708739778582739737422552531974179680294967

$d =$

74371802615146241472896188271041978689492703240379646141995046
 73820493661253184594168451370887921723607130820247798059973262
 06710810842150444333271043589416784775482885854461951

Exp. 6:

$N =$

23106605651041615213646000110077851420712757917776430930326056
 60296830951476238797402172985663226841689573922131931924256474
 42373164767688138199312178870617014197433167498889179888412544
 74185076959806015212320711881448073618683164329640885570487353
 50354182223829749002048540419684771479273577984952576597834431
 10876017808739917408013571983470008797535808485837831985030128
 18980694013921518439031616253599644305480276822960370194442400
 29680528535253331304066794571579472555287555198582878649996706
 70189200424088850932169339900976909686922394456370919349825147
 79933248467489575820613721302800274215988236180595306270631

$e =$

12306595799514227604950469560934424785746654745458886525393457
 58899783970499242533791938472818667104241589586924790854906048
 54925263611470499944927026328816440185366245189763792436116837
 57561103528778660481890339014924778781448952317743651644045890
 34734358672625085976939766863203987473622112403467008193246531
 18811333558134310761081974826858002018445896729330025874175254
 08511656625722972253589317139568494268155847524384162663775384
 56174150943940984854862543628588707580008442653426773237869939
 26533039846186871551451618391840130895864570824139236237941156
 12847782672209965094853232413711557861305091453331658286885

$p =$

15200857097888143612235928515088793346264155533550218885999558
 43118029224237682629271323091741303390999578531540748865895441
 28299767695057057649418113935154004959819709868376351089412151
 66689369221389374628670381926958230967689721241987218903900245
 6978031157591446602279278447377998069848764568730671548941379

$q =$

15200857097888130222480933300617571310430741971257868954229473
 99010773432783299974566489666130612156702118428957004402934567
 26259649410467624692928043525938992850157984384838932304533523
 37960458325033282161048364788519071913365063106030773291799585
 7967631386967340066303738923024637820931061972877004540650189

$d =$

95106665770280602013064006456747612094114888110214050993001292

18842148846561624186845878708855870619895801364016214375097533

53978037760812938848119029882054844495851034227001786365