Research Reports on Mathematical and Computing Sciences

Simplification of the lattice based attack of Boneh and Durfee for RSA cryptoanalysis

Yoshinori Aono

Department of
Mathematical and
Computing Sciences
Tokyo Institute of Technology

SERIES C: Computer Science

Simplification of the lattice based attack of Boneh and Durfee for RSA cryptoanalysis

Yoshinori Aono [*]

September 30, 2009

Abstract

In this paper we present a new formulation and its simpler analysis of the lattice based attack of Boneh and Durfee for the RSA cryptography [2]. We follow the same approach of Boneh and Durfee, however we propose a new way of defining a lattice with which we can achieve the same solvable key bound $d < N^{0.292}$. Our lattice is represented as a lower triangle matrix, which makes its analysis much simpler that of [2]. We think that this simpler analysis would be useful for considering applications/generalisations of this approach. In fact, as an example of such applications, we give a way of attacking RSA secret key with a certain repetitive structure.

1 Introduction

In [2], Boneh and Durfee proposed a polynomial time attack by which we can recover the RSA secret key d from the public information (e, N) when $d < N^{0.292}$; in the following, we call the bound $d < N^{0.292}$ the *solvable key bound* of Boneh and Durfee or simply the *Boneh-Durfee bound*. The basic idea of the attack is based on the Coppersmith technique by which we can obtain small solutions of a modular equation such as $f(x_1, x_2, \ldots, x_n) \equiv 0 \pmod{W}$. The technique converts the problem of finding a small solution of the equation to the problem of solving a system of polynomial equations by a lattice reduction algorithm such as the LLL algorithm [9].

Here is more detail explanation of their approach. The goal is to obtain a small solution (x_0, y_0) of the following target equation to recover the secret key.

$$f_{\mathrm{BD}}(x, y) = -1 + x(y + A) \equiv 1 \pmod{e} \tag{1}$$

Here $A = N + 1$. From this first the following bivariate polynomials are defined

$$g_{i,j}(x, y) = \begin{cases} x^{i-j}(f_{\mathrm{BD}}(x, y))^i e^{m-i} & \text{for } i \geq j \\ y^{j-i}(f_{\mathrm{BD}}(x, y))^j e^{m-j} & \text{for } i < j \end{cases} \tag{2}$$

for a certain range of (i, j) and an integer m. These polynomials are converted to a lattice represented by a row echelon matrix L_{BD} defined by using the coefficients of $g_{i,j}(x, y)$ with some parameters. Then by using a lattice reduction algorithm, we obtain a system of polynomial equations from which we can compute polynomial number of candidates of the solution (x_0, y_0) numerically.

In this approach a technically crucial point is to design a matrix for a lattice with a small determinant. They showed that their matrix has a sufficiently small determinant; however, its

---

[*]Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan, `aono5@is.titech.ac.jp`

analysis is complicated since the technique of geometrically progressive matrices, and it seems hard to apply for the other situations. The purpose of this paper is to give a new way to construct a lattice with asymptotically the same determinant that is much simpler to analyse.

Since Boneh and Durfee's work, variants of their technique have been proposed. Blömer and May [3] proposed a new lattice based algorithm for attacking RSA with a short secret key. They constructed a lower triangle lattice by eliminating some columns from the original lattice; this makes simplify the determinant analysis. In [8], Jochemsz and May gave an algorithm for finding small roots of a multivariate modular/integer equation based on a generalised lattice construction strategy. Note that both algorithms, achieve a slightly weaker solvable key bound than the Boneh-Durfee bound.

In this paper we follow the strategy of Boneh and Durfee to give a new variation of the lattice based attack with a simpler analysis. We propose a conversion from the polynomials (2) to three-variable polynomials $G_{i,j}(x, y, z)$ when we construct lattice; on the other hand, Boneh and Durfee directly constructed the lattice from $g_{i,j}(x, y)$. Since we obtain a lower triangle matrix representation of our lattice, we can easily compute its determinant. Therefore, we give a new simple algorithm to achieve the Boneh-Durfee bound.

We carry out our computer experiments to compare the qualities of our lattice and that of Boneh and Durfee. We check the solvable key ranges, the determinants, and the length of obtained vectors by L^2 algorithm [10, 11] on lattice generated by these two algorithms. As shown in Section 5, we confirm that the qualities of the two lattice series are equivalent for various parameters in practice. We find the computational time of the L^2 algorithm is reduced by about 30% from the original attack of Boneh and Durfee.

As the application of our analysis technique we consider the situation where an RSA secret key has a repetitive structure; more precisely, the situation that the secret key d (in its binary representation) is the repeat of r short bit string d_0. In this situation we can reduce the problem of recovering secret key to that of finding small solution pairs of

$$f_{\mathrm{rep}}(x, y) = -1 + x(y + A) \equiv 1 \pmod{eR}$$

where $R = 1 + 2^\ell + \cdots + 2^{(r-1)\ell}$; see Section 6 for precise information. For this case we can recover the secret key when

$$\beta < \frac{r + 3 - \sqrt{r^2 + 6r + 1}}{4}$$

where $\beta = \log_N d_0$. For $r = 1$, we can see this is equivalent to the attack of Boneh and Durfee.

This paper is organised as follows: In section 2, we give basic symbols, notations, lemmas. We give our formulation of the lattice based attack in section 3 and its detailed analysis is explained in 4. The computer experiments to compare our lattice construction and that in [2] is described in section 5. In section 6, we demonstrate that our approach can be used to analyse the situation when a secret key has a repetitive structure.

2   Preliminaries

In this section, for the following discussions, we introduce some notations, state some known facts, and key technical lemmas.

We use standard RSA notations throughout this paper. A given RSA instance is defined by p, q, e, and d, where p and q are large primes, e is a public key, and d is the corresponding secret

key. Let $N = p \times q$, and let $\varphi(N)$ be the Euler's function; here we may simply assume that $\varphi(N) = (p-1)(q-1)$. We assume that $\gcd(e, \varphi(N)) = 1$. The key relation between e and d is

$$ed \equiv 1 \pmod{\varphi(N)} \tag{3}$$

from which we derive our target equation (1) by following the argument in [2].

The basic strategy of the lattice based attack is to convert the problem of recovering RSA to the problem of finding small solution of a modular equation; more precisely, this problem is to find a solution within a certain range of a modular equation such as $f(x, y) \equiv 0 \pmod{W}$ for a polynomial $f(x, y)$ and a nonnegative integer W. In general, solving modular equation is not easy, whereas there are some cases where we may be able to use the standard numerical method for solving this problem. The Howgrave-Graham lemma [7] provides us with one of such cases.

To state the Howgrave-Graham lemma, we introduce the following norm for bivariate polynomials and integers.

Definition 1. XY**-norm**   Let $f(x, y) = \sum_{i,j} a_{i,j} x^i y^j$ be a polynomial with integral coefficients, X and Y be natural numbers. We define the XY-norm of $f(x, y)$ by

$$\|f(x, y)\|_{XY} \stackrel{\text{def}}{=} \sqrt{\sum_{i,j} a_{i,j}^2 X^{2i} Y^{2j}}.$$

Lemma 1. (Howgrave-Graham [7]) *For any positive integers X, Y and W, let $f(x, y)$ be a bivariate polynomial consisting with w terms with integral coefficients such that the following holds*

$$\|f(x, y)\|_{XY} < \frac{W}{\sqrt{w}}.$$

Then we have

$$f(x, y) \equiv 0 \pmod{W} \Leftrightarrow f(x, y) = 0$$

within the range of $|x| < X$ and $|y| < Y$.

Note that $f(x, y) = 0$ clearly implies $f(x, y) \equiv 0 \pmod{W}$. What is important is its converse. This lemma guarantees that we can find all solutions of the modular equation within the range from the integral solutions of $f(x, y) = 0$ (if they exist).

Now we introduce some definitions and some lemmas about the lattice; we need to obtain a polynomial with a small XY-norm to use Lemma 1, and this problem can be reduced to a problem of finding a short vector in a lattice. Consider linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$, then the lattice with basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is defined by

$$L(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i \,\middle|\, a_i \in \mathbb{Z} \text{ for } i = 1, \ldots, n \right\}. \tag{4}$$

That is, the lattice is the set of integral linear combinations of its basis vectors.

The shortest vector problem, for given basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$, is to find a vector $\mathbf{v}$ such that

- $\mathbf{v} \in L(\mathbf{b}_1, \ldots, \mathbf{b}_n) \setminus \{\mathbf{0}\}$ and
- $|\mathbf{v}| \leq |\mathbf{v}'|$ for $\forall \mathbf{v}' \in L(\mathbf{b}_1, \ldots, \mathbf{b}_n) \setminus \{\mathbf{0}\}$.

In other word, this problem is to find a non-zero vector having the minimum length in $L(\mathbf{b}_1, \ldots, \mathbf{b}_n)$. We know that a lattice basis reduction algorithm finds a good approximation of the problem by computing a reduced basis. We use the LLL algorithm [9], the most widely used lattice reduction algorithm, in our analysis. The two short vectors in the reduced basis described in the following theorem are important.

Theorem 1. *[2, Fact 3.3] Let $\mathbf{b}_1, \ldots, \mathbf{b}_n$ be a given linearly independent basis. Then the LLL algorithm can find linearly independent lattice vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ such that*

$$|\mathbf{v}_1| \leq 2^{(n-1)/4} |\det(L)|^{1/n} \text{ and}$$
$$|\mathbf{v}_2| \leq 2^{n/2} |\det(L)|^{1/(n-1)}. \tag{5}$$

Here, $\det(L)$ is the determinant of the lattice which is defined by the determinant of a matrix representation of the lattice; this is actually defined by

$$\det(L) = \det \begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_n \end{pmatrix}.$$

When we obtain a lower triangle matrix representation of a lattice, we can easily compute $|\det(L)|$ by the product of its diagonal elements. we convert the short vectors in the reduced basis to polynomials satisfying the sufficient condition of the Howgrave-Graham lemma.

We introduce a mapping for converting polynomials to vectors; since a lattice reduction algorithm is designed for vectors, while our targets are polynomials. We divide this mapping into two steps, named a vectorisation and an instantiation respectively. We introduce a way to map three-variable polynomials to vectors since we will consider three-variable polynomials in our construction. Our mapping is natural and simple; for example the polynomial $f(x, y, z) = -3x^3 + 4x^2yz - 2xy^2z^2 + 7xy^3z^3$ is mapped to the vector $(-3x^3, 4x^2yz, -2xy^2z^2, 7xy^3z^2)$ by following a fixed linear order of monomials. We state formally this mapping as follows.

Definition 2. Polynomials $\Rightarrow$ vectors

Let $\mathbf{K}$ be a finite sequence of distinct three-variable monomials. We assume a linear order on this, and let it be fixed; for any t, let $x^{i_t} y^{j_t} z^{k_t}$ be the t-th monomial in this order. Then for any $f(x, y, z) = \sum_{1 \leq t \leq |\mathbf{K}|} a_t x^{i_t} y^{j_t} z^{k_t}$, we map it to the following vector $\mathbf{b}$, which is called the vectorisation of $f(x, y, z)$ and is denoted as $\mathcal{V}_{\mathbf{K}}(f)$.

$$
\begin{array}{ccccccccc}
f(x, y, z) & = & a_1 x^{i_1} y^{j_1} z^{k_1} & + & a_2 x^{i_2} y^{j_2} z^{k_2} & + & \cdots & + & a_{|\mathbf{K}|} x^{i_{|\mathbf{K}|}} y^{j_{|\mathbf{K}|}} z^{k_{|\mathbf{K}|}} \\
 & & \downarrow & & \downarrow & & & & \downarrow \\
\mathbf{b} & = (& a_1 x^{i_1} y^{j_1} z^{k_1} & , & a_2 x^{i_2} y^{j_2} z^{k_2} & , & \ldots & , & a_{|\mathbf{K}|} x^{i_{|\mathbf{K}|}} y^{j_{|\mathbf{K}|}} z^{k_{|\mathbf{K}|}} &).
\end{array}
$$

We introduce a conversion named an instantiation and its inverse; it converts a three-variable monomials to integers by substitution. Our matrix will be defined by using the vectorizations and hence each element of the matrix is monomial. On the other hand, a lattice reduction algorithm is designed for integer lattices or integer matrices. Thus, for using a lattice reduction algorithm, we need to instantiate our matrix by substituting some integers X, Y and Z to x, y and z, which we call an *instantiation* with X, Y and Z. Conversely, converting an integer vector to a polynomial is called a *deinstantiation*. Note that (since $\mathbf{K}$ and the order of monomials is fixed) we know a monomial $x^{i_t} y^{j_t} z^{k_t}$ corresponding to the t-th entry of a given vector; hence, deinstantiation at the t-th entry can be achieved by simply dividing its integral value by $X^{i_t} Y^{j_t} Z^{k_t}$.

These vectorization, instantiation, and deinstantiation procedures are essentially the same as those used by Boneh and Durfee (except that we consider three-variable polynomials while bivariate polynomials have been used by them).

3 A New Lattice Based Algorithm

In this section we give a new lattice based algorithm for RSA with a short secret key; that is, a new lattice construction and its simpler analysis to derive the same Boneh-Durfee bound $d < N^{0.292}$. Our analysis requires only elementary lemmas and calculations. The detailed analysis is given in the next section. What is different from the original algorithm is to use three-variable polynomials to construct a lattice; the bivariate polynomials (2) are used directly in the original paper. We first state some definitions and lemma to explain our lattice construction.

Canonical Replacement

We convert the bivariate polynomials $g_{i,j}(x,y)$ to three-variable polynomials artificially. We first express $g_{i,j}(x,y)$ as a sum of monomials and then replace every xy by $z+1$ in $g_{i,j}(x,y)$. For example, the polynomial $g_{2,3}(x,y) = e^{m-2}(-1+xy+Ax)^2y = e^{m-2}y + A^2e^{m-2}x^2y - 2Ae^{m-2}xy - 2e^{m-2}xy^2 + 2Ae^{m-2}x^2y^2 + x^2y^3$ is converted to the polynomial $G_{3,2}(x,y,z) = e^{m-2}y + A^2e^{m-2}x(1+z) - 2Ae^{m-2}(1+z) - 2e^{m-2}y(1+z) + 2Ae^{m-2}(1+z)^2 + e^{m-2}y(1+z)^2$. Like this example, we will denote the converted polynomial from $g_{i,j}(x,y)$ by $G_{i,j}(x,y,z)$, and we call this conversion a *canonical replacement*. It is clear that $G_{i,j}(x,y,-1+xy) = g_{i,j}(x,y)$. Though artificial, this canonical replacement allows us to define a lower triangle matrix representation of our lattice.

Here we extend the notion of XY-norm for converted three-variable polynomials and show some useful bound. Though we consider such converted three-variable polynomials, they are essentially bivariate polynomials; hence, we still discuss its XY-norm and show an inequality. Let $F(x,y,z) = \sum_{i,j,k} b_{i,j,k}x^iy^jz^k$ be a three-variable polynomial. For this F, we define a three-variable version of the XY-norm as follows:

$$||F(x,y,z)||_{XY} \stackrel{\text{def}}{=} \sqrt{\sum_{i,j,k} b_{i,j,k}^2 X^{2i}Y^{2j}(X^2Y^2+1)^k}.$$

Again this definition is somewhat artificial; one motivation is to have the following bound.

Lemma 2. *Let $f(x,y)$ be any bivariate polynomial and let $F(x,y,z)$ be is a polynomial obtained by applying the canonical replacement to $f(x,y)$. Let v be the maximum degree of z in $F(x,y,z)$. Then for any non-negative integers X and Y, the following holds.*

$$||f(x,y)||_{XY} \le (v+1)||F(x,y,z)||_{XY}.$$

Proof. We let $f(x,y) = \sum_{s,t} a_{s,t}x^sy^t$, and let $F(x,y,z) = \sum_{i,j,k} b_{i,j,k}x^iy^jz^k$. Then since F is obtained from f by the canonical replacement, it follows that $f(x,y) = F(x,y,-1+xy) = \sum_{i,j,k} b_{i,j,k}x^iy^j(-1+xy)^k$. We give the relationship between $a_{s,t}$'s and $b_{i,j,k}$'s as follows.

$$\sum_{i,j,k} b_{i,j,k}x^iy^j(-1+xy)^k = \sum_{i,j,k} b_{i,j,k}x^iy^j \sum_{c=0}^{k} \binom{k}{c}(-1)^{k-c}x^cy^c$$

$$= \sum_{i,j,k,c} b_{i,j,k}\binom{k}{c}(-1)^{k-c}x^{i+c}y^{j+c} = \sum_{s,t,k,c} b_{s-c,t-c,k}\binom{k}{c}(-1)^{k-c}x^sy^t.$$

Here, we let $s = i + c$ and $t = j + c$. By comparing the coefficient of x^sy^t, we have

$$a_{s,t} = \sum_{k,c} b_{s-c,t-c,k}(-1)^{k-c}\binom{k}{c}. \tag{6}$$

Now k and c are integers between 0 and v since its range is from the degree of z in $F(x, y, z)$. Hence, the number of terms in the right-hand side of (6) is equal to or less than $(v+1)^2$. Thus we have

$$|a_{s,t}|^2 \le (v+1)^2 \sum_{k,c} \left| b_{s-c,t-c,k} \binom{k}{c} \right|^2.$$

By this inequality, we derive our claim as follows:

$$
\begin{aligned}
||f(x,y)||_{XY}^2 &= \sum_{s,t} |a_{s,t}|^2 X^{2s} Y^{2t} \le \sum_{s,t}(v+1)^2 \sum_{k,c} \left| b_{s-c,t-c,k} \binom{k}{c} \right|^2 X^{2s} Y^{2t} \\
&= (v+1)^2 \sum_{s,t,k,c} \left| b_{s-c,t-c,k} \binom{k}{c} \right|^2 X^{2s} Y^{2t} \\
&= (v+1)^2 \sum_{i,j,k,c} |b_{i,j,k}|^2 \binom{k}{c}^2 X^{2(i+c)} Y^{2(j+c)} \\
&= (v+1)^2 \sum_{i,j,k} |b_{i,j,k}|^2 X^{2i} Y^{2j} \sum_{c=0}^{k} \binom{k}{c}^2 X^{2c} Y^{2c} \\
&= (v+1)^2 \sum_{i,j,k} |b_{i,j,k}|^2 X^{2i} Y^{2j} (1 + X^2 Y^2)^k \\
&= (v+1)^2 ||F(x,y,z)||_{XY}^2.
\end{aligned}
$$

Thus we have $||f(x,y)||_{XY} \le (v+1)||F(x,y,z)||_{XY}$. $\square$

Remark. We will assume that $Z = \sqrt{X^2 Y^2 + 1}$ whenever we consider instantiation/deinstantiation with some X and Y to keep its consistent with this extended XY-norm notion. Thus, for any three-variable polynomial $F(x,y,z)$ obtained as a sum of monomials of the deinstantiation of some vector $\mathbf{F}$ w.r.t. X and Y, the following relation is immediate.

$$||F(x,y,z)||_{XY} = |\mathbf{F}|. \tag{7}$$

Now we explain our version of the lattice based attack for RSA following its outline stated in Figure 1. This is essentially the same as the one by Boneh and Durfee except for polynomials and a lattice construction.

We first define symbols used in the algorithm. Let δ be the ratio of the bit-length of d to that of N; here we assume that $\delta < 0.5$. Let m be an another parameter that is set as an integer greater than one; the larger m would yield the better solvable key range but the more computation time is necessary. The Boneh-Durfee bound $\delta < 0.292$ is the approximated value when we take sufficiently large m. Thus, considering available computational resource and δ, an appropriate number should be chosen for m; for our experiment, we set m from 6 to 10.

Then we define the set $I = \{(i,j) \in \mathbb{Z}^2 | 0 \le i \le m, 0 \le j \le 2(1-\delta)i\}$. The sequence $\mathbf{I}$ is defined by introducing some order to elements in I; however we postpone its explanation to the next section. For $(i,j) \in I$, polynomials $g_{i,j}(x,y)$ are defined as follows:

$$g_{i,j}(x,y) \stackrel{\text{def}}{=} \begin{cases} x^{i-j}(f_{\text{BD}}(x,y))^i e^{m-i} & \text{for } i \ge j \\ y^{j-i}(f_{\text{BD}}(x,y))^j e^{m-j} & \text{for } i < j. \end{cases} \tag{8}$$

6

Step 1:	Choose attack parameters m and δ.
Step 2:	Define an index sequence $\mathbf{I}$ and a monomial sequence $\mathbf{K}$ (as explained in Section 4). For each $(i,j) \in \mathbf{I}$, define a polynomial $g_{i,j}(x,y)$ as (2) and a polynomial $G_{i,j}(x,y,z)$ by the canonical replacement of $g_{i,j}(x,y)$. Construct a lattice L using vectors $\mathcal{V}_{\mathbf{K}}(G_{i,j})$ as row vectors in the order of (i,j) following $\mathbf{I}$.
Step 3:	Instantiate L with $X = \lfloor N^\delta \rfloor$, $Y = \lfloor N^{0.5} \rfloor$ (and $Z = \sqrt{X^2Y^2+1}$). Then apply a lattice reduction algorithm to it.
Step 4:	For two short vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ computed by a reduction algorithm, compute their deinstantiations $\mathbf{v}_1'$ and $\mathbf{v}_2'$. Define polynomials $H_1(x,y,z)$ and $H_2(x,y,z)$ by summing up the monomials in $\mathbf{v}_1'$ and $\mathbf{v}_2'$ respectively. Then define $h_1(x,y) = H_1(x,y,-1+xy)$ and $h_2(x,y) = H_2(x,y,-1+xy)$.
Step 5:	Enumerate all integral solutions of $h_1(x,y) = h_2(x,y) = 0$. For each of those solutions, compute d by (1) and check whether it is an integer.

Figure 1: Our version of the lattice based attack

These are the same polynomials defined in [2]; more precisely, our $g_{i,j}(x,y)$ for $i \geq j$ and for $i < j$ correspond to their $g_{i,j}(x,y)$ and $h_{i,j}(x,y)$ respectively. We then further extend them to three-variable polynomials $G_{i,j}(x,y,z)$ by the canonical replacement. Consider the set of monomials of type $x^i y^j z^k$ that appear in some $G_{i,j}(x,y,z)$. Again its ordered version $\mathbf{K}$ will be defined in the next section. Now we let $\mathbf{b}_{i,j} = \mathcal{V}_{\mathbf{K}}(G_{i,j})$ and define our lattice L as follows:

$$L = \begin{bmatrix} \mathbf{b}_{0,0} \\ \mathbf{b}_{0,1} \\ \vdots \\ \mathbf{b}_{m,m'} \end{bmatrix} \tag{9}$$

where m' is $\lfloor 2(1-\delta)m \rfloor$. One important point here is that we can choose some appropriate ordering for $\mathbf{K}$ so that L becomes lower triangle; we prove this in the next section.

Next we carry out a lattice reduction algorithm on L' that is obtained as the instantiation of L with parameters $X = \lfloor N^\delta \rfloor$ and $Y = \lfloor 3N^{0.5} \rfloor$ (and $Z = \sqrt{1 + X^2Y^2}$); for our analysis, we consider the LLL algorithm. The algorithm computes a reduced basis, which contains short vectors in the lattice.

From two short vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ in the reduced basis, we construct the corresponding polynomials $h_1(x,y)$ and $h_2(x,y)$. First we convert the two vectors to their deinstantiations $\mathbf{v}_1'$ and $\mathbf{v}_2'$. Then define $H_1(x,y,z)$ and $H_2(x,y,z)$ as the sums of all monomials in $\mathbf{v}_1'$ and $\mathbf{v}_2'$ respectively. From the construction of L and the nature of the instantiation/deinstantiation, that is, we can easily see each $\mathbf{v}_c$ is a integral linear combination of $\mathbf{b}_i$'s; hence this yields that $H_1(x,y,z)$ and $H_2(x,y,z)$ are integral linear combinations of $G_{i,j}(x,y,z)$. We then obtain $h_1(x,y)$ and $h_2(x,y)$ by $h_c(x,y) = H_c(x,y,-1+xy)$ for $c = 1, 2$.

Finally, in step 5, we solve the simultaneous equation $h_1(x,y) = h_2(x,y) = 0$. For each (x_1, y_1) of the integral solutions of the equation, compute d as follows

$$d = \frac{-1 + x_1(y_1 + A)}{e}$$

and check it is indeed the correct secret key, i.e., check whether it is a non-negative integer. This is the outline of our version of the lattice based attack.

Now we show relationships between the polynomials computed in the algorithm; our target is to derive the Boneh-Durfee bound via considering a sufficient condition of the Howgrave-Graham lemma. By construction we have

$$f_{\text{BD}}(x,y) \equiv 0 \pmod{e} \Rightarrow [g_{i,j}(x,y) = G_{i,j}(x,y,-1+xy) \equiv 0 \pmod{e^m} \text{ for } {}^\forall (i,j) \in I].$$

On the other hand, $[G_{i,j}(x,y,-1+xy) \equiv 0 \pmod{e^m} \text{ for } {}^\forall (i,j) \in I] \Rightarrow [H_c(x,y,-1+xy) = h_c(x,y) \equiv 0 \pmod{e^m} \text{ for } c = 1,2]$ since each $H_c(x,y,z)$ is an integral linear combination of $G_{i,j}(x,y,z)$. Thus, if both $h_1(x,y)$ and $h_2(x,y)$ satisfy the condition of the Howgrave-Graham lemma for X, Y and $W = e^m$, we have

$$h_c(x,y) \equiv 0 \pmod{e^m} \Leftrightarrow h_c(x,y) = 0 \text{ for } c = 1,2, \; |x| < X \text{ and } |y| < Y$$

and this implies

$$f_{\text{BD}}(x,y) \equiv 0 \pmod{e} \Rightarrow h_c(x,y) = 0 \text{ for } c = 1,2, \; |x| < X \text{ and } |y| < Y. \tag{10}$$

Therefore a small solution of $f_{\text{BD}}(x,y) \equiv 0 \pmod{e}$ must be included in the set of small solutions of $h_1(x,y) = h_2(x,y) = 0$ when h_1 and h_2 satisfy the Howgrave-Graham condition; more precisely, if for each $c = 1,2$, h_c satisfies the following.

$$||h_c(x,y)||_{XY} < \frac{e^m}{\sqrt{w}} \tag{11}$$

Here w is the number of terms in $h_c(x,y)$, which is equal or less than $(1-\delta)m^2$.

Now we consider this condition for each $h_c(x,y)$ to derive the Boneh-Durfee bound. We have by Lemma 2,

$$||h_c(x,y)||_{XY} \leq (v+1)||H_c(x,y,z)||_{XY} \leq (m+1)||H_c(x,y,z)||_{XY}$$

where v is the maximum degree of z in $H_c(x,y,z)$, which is equal to or smaller than m. Moreover by Theorem 1 and (7) we have

$$||H_c(x,y,z)||_{XY} = |\mathbf{v}_c| \leq 2^{n/2} \det(L')^{1/(n-1)}.$$

Rearranging these conditions, we have a sufficient condition for the Howgrave-Graham lemma as follows.

$$(m+1)2^{n/2}\det(L')^{1/(n-1)} < \frac{e^m}{\sqrt{1-\delta m}}.$$

Following the analysis of Boneh and Durfee, we disregard the numbers $(m+1)2^{n/2}$ and $\sqrt{1-\delta m}$ because these are sufficiently small comparing to the RSA parameters, and we use $\det(L')^{1/n}$ instead of $\det(L')^{1/(n-1)}$. Hence we have our simplified sufficient condition.

$$\det(L')^{1/n} < e^m. \tag{12}$$

Here from the analysis of the next section, we have both

$$n = (1-\delta)m^2 + o(m^2)$$

and

$$\det(L') = N^{\left(-\frac{1}{3}\delta^2 - \frac{1}{3}\delta + \frac{5}{6}\right)m^3 + o(m^3)}.$$

8

Therefore, the condition (12) is equivalent to

$$N^{\left(-\frac{1}{3}\delta^2 - \frac{1}{3}\delta + \frac{5}{6}\right)m^3 + o(m^3)} = \det(L') < e^{nm} = N^{(1-\delta)m^3 + o(m^3)}.$$

Hence we have

$$-\frac{1}{3}\delta^2 - \frac{1}{3}\delta + \frac{5}{6} < 1 - \delta \Leftrightarrow 2\delta^2 - 4\delta + 1 < 0.$$

for sufficiently large m. Then we have the condition for δ as follows

$$\delta < 1 - \frac{1}{\sqrt{2}} \approx 0.292. \tag{13}$$

This is the same as the Boneh-Durfee bound [2].

4 Analysis in Detail

In this section we show that L defined in the above section is lower triangle; hence we can easily derive the determinant of the L' defined as the instantiation of L with parameters X and Y.

We need to give the detailed construction of L to prove our claim; before this, we define an index sequence $\mathbf{I}$ and a monomial sequence $\mathbf{K}$ to set an order of terms in our matrix. For fixed m and $\delta < 0.5$, we define the set $I \stackrel{\text{def}}{=} \{(i,j) \in \mathbb{Z}^2 | 0 \leq i \leq m, 0 \leq j \leq 2(1-\delta)i\}$. We respectively define $\mathbf{I}_1$ and $\mathbf{I}_2$ by the lexicographic order of (i,j) in $\{(i,j) \in I | i \geq j\}$ and that of (j,i) in $\{(i,j) \in I | i < j\}$; we use these sequences to define the order of the vector $G_{i,j}(x,y,z)$. We further set the index sequence $\mathbf{I}$ as the concatenation of $\mathbf{I}_1$ and $\mathbf{I}_2$. For $\mathbf{I_1} = ((i_1, j_1), \ldots, (i_u, j_u))$ and $\mathbf{I_2} = ((i'_1, j'_1), \ldots, (i'_{u'}, j'_{u'}))$, we construct monomial sequences $\mathbf{K}_1$ and $\mathbf{K}_2$; we use these sequences to set the monomial order in vectorization. We define the monomial sequences $\mathbf{K}_1$ and $\mathbf{K}_2$ by $\mathbf{K_1} = (x^{i_1-j_1}z^{j_1}, \ldots, x^{i_u-j_u}z^{j_u})$ and $\mathbf{K_2} = (y^{j'_1-i'_1}z^{i'_1}, \ldots, y^{j'_{u'}-i'_{u'}}z^{i'_{u'}})$ respectively. We also set $\mathbf{K}$ by the concatenation of $\mathbf{K}_1$ and $\mathbf{K}_2$. We use these sequences to define our L. Here we give a small example for $m = 3$ and $\delta = 0.25$; we have $\mathbf{I_1} = ((0,0),(1,0),(1,1),(2,0), (2,1),(2,2),(3,0),(3,1),(3,2),(3,3))$ and $\mathbf{I_2} = ((2,3),(3,4))$. By them, we have the monomial sequence $\mathbf{K_1} = (1, x, z, x^2, xz, z^2, x^3, x^2z, xz^2, z^3)$ and $\mathbf{K_2} = (yz^2, yz^3)$.

We state two facts for our analysis; we will use them in the proof of Lemma 3 and Lemma 4. We denote a symbol $\prec$ the order in $\mathbf{K}_1$ and $\mathbf{K}_2$.

Fact 1. *We have for the elements in $\mathbf{K}_1$, $x^i z^j \prec x^{i'} z^{j'} \Leftrightarrow i + j < i' + j'$ or $[i + j = i' + j'$ and $j < j']$. For the elements in $\mathbf{K}_2$, $y^i z^j \prec y^{i'} z^{j'} \Leftrightarrow i + j < i' + j'$ or $[i + j = i' + j'$ and $j < j']$.*

Fact 2. *For elements in $\mathbf{K}$, we have $x^j z^i \in \mathbf{K}_1 \Leftrightarrow 0 \leq i + j \leq m$, and $y^j z^i \in \mathbf{K}_2 \Leftrightarrow [0 \leq i \leq m$ and $0 < j < (1 - 2\delta)i]$*

Now we define our lattice L by using the polynomials $G_{i,j}(x,y,z)$ and the defined sequences; here we actually give a matrix representation of L. Our matrix is defined by the row matrix of vectors $\mathcal{V}_\mathbf{K}(G_{i,j})$ for $(i,j) \in I$ whose order is from $\mathbf{I}$. We divide L as follows to show its lower triangularity:

$$L = \begin{bmatrix} \mathcal{V}_{\mathbf{K}}(G_{0,0}) \\ \vdots \\ \mathcal{V}_{\mathbf{K}}(G_{m,m'}) \end{bmatrix} = \begin{bmatrix} \overbrace{L_{00}}^{\mathbf{K}_1} & \overbrace{L_{01}}^{\mathbf{K}_2} \\ \hline L_{10} & L_{11} \end{bmatrix} \begin{matrix} \left.\vphantom{L_{00}}\right\}\mathbf{I}_1 \\ \left.\vphantom{L_{10}}\right\}\mathbf{I}_2 \end{matrix} \tag{14}$$

Here, $m' = \lfloor 2(1-\delta)m \rfloor$. Therefore, we need to show that L_{00} and L_{11} are lower triangle matrices and show that L_{01} is the zero matrix to proof the triangularity of L; of course, we need to prove that the monomials in $G_{i,j}(x,y,z)$ are contained in $\mathbf{K}$. This will be showed via the proof of Lemma 3 and Lemma 4.

Lemma 3. *L_{00} and L_{01} are a lower triangle matrix and the zero matrix respectively.*

Proof. Let (i_k, j_k) be k-th element in $\mathbf{I}_1$. We first show the triangularity of L_{00}; we need to show that the polynomial $G_{i,j}(x,y,z)$ can be expressed as an linear combination of the first k elements in $\mathbf{K}_1$, and show that the coefficient of $x^{i_k-j_k}z^{j_k}$ in $G_{i_k,j_k}(x,y,z)$ is not zero.

The expression of $G_{i,j}(x,y,z)$ is computed as follows by the definition (8) and the canonical replacement:

$$\begin{aligned} G_{i_k,j_k}(x,y,z) &= x^{i_k-j_k}(-1+xy+Ax)^{j_k}e^{m-j_k} \\ &= x^{i_k-j_k}(z+Ax)^{j_k}e^{m-j_k} = \sum_{\ell=0}^{j_k} a_\ell x^{i_k-\ell}z^\ell. \end{aligned}$$

where a_ℓ are certain integers.

Thus, by the Fact 1, $x^{i_k-j_k}z^{j_k}$, this is the k-th element in $\mathbf{K}_1$, is the most right non-zero element in $\mathcal{V}_{\mathbf{K}}(G_{i_k,j_k})$ in the order $\prec$; this also corresponds to the k-th diagonal element in our matrix. Hence the non-zero elements in $\mathbf{b}_{i_k,j_k}$ are on the diagonal position or its left; This shows that L_{00} is a lower triangle matrix. It is clear that L_{01} is the zero-matrix since the polynomial $G_{i_k,j_k}(x,y,z)$ for $(i_k,j_k) \in \mathbf{I}_1$ does not have a monomial of type $z^{j'}y^{i'}$. $\square$

Lemma 4. *L_{11} is a lower triangle matrix.*

Proof. Let (i_k, j_k) be the k-th element in $\mathbf{I}_2$, We carry the proof by showing the monomials $G_{i_k,j_k}(x,y,z)$ include in $\mathbf{K}_1$ and the first k elements in $\mathbf{K}_2$.

We first give an expression of $G_{i_k,j_k}(x,y,z)$ for $(i_k,j_k) \in \mathbf{I}_2$; we have by (8) and the canonical replacement,

$$G_{i_k,j_k}(x,y,z)/e^{m-i_k} = y^{j_k-i_k}(-1+xy+Ax)^{i_k} = y^{j_k-i_k}(z+Ax)^{i_k}$$

$$= \sum_{t=0}^{i_k} \binom{i_k}{t}(Ax)^t y^{j_k-i_k} z^{i_k-t}$$

$$= \sum_{t=0}^{j_k-i_k-1} \binom{i_k}{t}(Ax)^t y^{j_k-i_k} z^{i_k-t} + \sum_{t=j_k-i_k}^{i_k} \binom{i_k}{t}(Ax)^t y^{j_k-i_k} z^{i_k-t}$$

$$= \sum_{t=0}^{j_k-i_k-1} \binom{i_k}{t}A^t(xy)^t y^{j_k-i_k-t} z^{i_k-t} + \sum_{t=j_k-i_k}^{i_k} \binom{i_k}{t}A^t(xy)^{j_k-i_k} x^{t-j_k+i_k} z^{i_k-t}$$

$$= \sum_{t=0}^{j_k-i_k-1} \binom{i_k}{t}A^t(1+z)^t y^{j_k-i_k-t} z^{i_k-t} + \sum_{t=j_k-i_k}^{i_k} \binom{i_k}{t}A^t(1+z)^{j_k-i_k} x^{t-j_k+i_k} z^{i_k-t}$$

$$= \sum_{t=0}^{j_k-i_k-1} \binom{i_k}{t}A^t \sum_{\ell=0}^{t}\binom{t}{\ell} y^{j_k-i_k-t} z^{i_k-t+\ell} + \sum_{t=j_k-i_k}^{i_k} \binom{i_k}{t}A^t \sum_{\ell=0}^{j_k-i_k}\binom{t}{\ell} x^{t-j_k+i_k} z^{i_k-t+\ell}.$$

Therefore, we obtain that the monomials included in the expression of $G_{i_k,j_k}(x,y,z)$ are

$$x^{t-j_k+i_k} z^{i_k-t+\ell} \text{ for } j_k-i_k \le t \le i_k \text{ and } 0 \le \ell \le j_k-i_k, \tag{15}$$

and

$$y^{j_k-i_k-t} z^{i_k-t+\ell} \text{ for } 0 \le t \le j_k-i_k-1 \text{ and } 0 \le \ell \le t. \tag{16}$$

We show that the terms in (15) and (16) are included in $\mathbf{K}_1$ and $\mathbf{K}_2$ respectively. First we argue the terms in (15). By Fact 2, we have $x^{t-j_k+i_k} z^{i_k-t+\ell} \in \mathbf{K}_1 \Leftrightarrow 0 \le (t-j_k+i_k) + (i_k-t+\ell) = 2i_k - j_k + \ell \le m$, thus we need to show

$$0 \le 2i_k - j_k + \ell \le m \text{ for } (i_k,j_k) \in \mathbf{I}_2 \text{ and } 0 \le \ell \le j_k-i_k.$$

We have that $j_k-i_k \le t \le i_k$ derives $0 \le t-j_k+i_k \le 2i_k-j_k$. Then by $0 \le \ell \le j_k-i_k$, we have $0 \le 2i_k - j_k + \ell \le i_k \le m$. Hence, the monomials (15) are in $\mathbf{K}_1$.

Next we show that the terms (16) are the elements of $\mathbf{K}_2$ by similar argument. By Fact 2, we have

$$y^{j_k-i_k-t} z^{i_k-t+\ell} \in \mathbf{K}_2 \Leftrightarrow [0 \le i_k-t+\ell \le m \text{ and } 0 < j_k-i_k-t < (1-2\delta)(i_k-t+\ell)].$$

Hence we show that these two inequalities satisfy for $(i_k,j_k) \in \mathbf{I}_2$, $0 \le t \le j_k-i_k-1$ and $0 \le \ell \le t$. We have

$$i_k-t+\ell \le i_k \le m \text{ from } 0 \le \ell \le t.$$

We also have

$$i_k-t+\ell \ge 2i_k-j_k+1 > 2i_k-2(1-\delta)i_k+1 = 2\delta i_k+1 \ge 0$$

from $t \le j_k-i_k-1$ and $\ell \ge 0$. Thus we have $0 \le i_k-t+\ell \le m$. This is the first inequality. Next we show the second inequality. We have that $t \le j_k-i_k-1$ derives $j_k-i_k-t \ge 1 > 0$. On the other hand, $j_k-i_k-t < (1-2\delta)i_k-t \le (1-2\delta)i_k-t+\ell$ holds. Then we have for $0 \le \delta < 0.5$, $-t+\ell \le (1-2\delta)(-t+\ell)$ since $-t+\ell \le 0$. Hence we have

$$j_k-i_k-t < (1-2\delta)i_k-t+\ell \le (1-2\delta)i_k+(1-2\delta)(-t+\ell) \le (1-2\delta)(i_k-t+\ell).$$

Therefore, we have the monomials (16) are in $\mathbf{K}_2$.

We need to show each maximum element in (16) in the order $\prec$ corresponds to a diagonal element in L_{11}. We have by Fact 1, the maximum element in (16) is a monomial $y^{j_k - i_k - t} z^{i_k - t + \ell}$ such that:

$$(j_k - i_k - t) + (i_k - t + \ell) = j_k + \ell - 2t \text{ is maximum, and}$$
$$j_k - i_k - t \text{ is also maximum under maximized } j_k + \ell - 2t$$
$$\text{within the range of (16).}$$

Hence this is the case of $\ell = t = 0$; this corresponds the monomial $y^{j_k - i_k} z^{i_k}$, which is the k-th element in $\mathbf{K}_2$, hence, this is also the k-th diagonal element in L_{11}. Therefore L_{11} is a lower triangle matrix. $\square$

Now we can easily compute the determinant of L'; since L and its instantiation L' are lower triangle matrices by combining Lemma 3 and Lemma 4. We have from the expressions, the diagonal elements in L' corresponding to $G_{i,j}(x, y, z)$ are $e^{m-j} X^{i-j} Z^j$ for $(i, j) \in \mathbf{I}_1$, and $e^{m-i} Y^{j-i} Z^i$ for $(i, j) \in \mathbf{I}_2$ respectively. Hence by using the approximations $e \approx N, X \approx N^{\delta}, Y \approx N^{0.5}$ and $Z = \sqrt{X^2 Y^2 + 1} \approx N^{\delta + 0.5}$, we have

$$
\begin{aligned}
\det(L_{00}) &= e^{m(m+1)(m+2)/3} X^{m(m+1)(m+2)/6} Y^{m(m+1)(m+2)/6} \\
&= N^{\left(\frac{5}{12} + \frac{1}{3}\delta\right)m^3 + o(m^3)} \text{ and} \\
\det(L_{11}) &= e^{(1-2\delta)m^3/6 + o(m^3)} Y^{(1-2\delta)^2 m^3/6 + o(m^3)} Z^{(1-2\delta)m^3/3 + o(m^3)} \\
&= N^{\left(-\frac{1}{3}\delta^2 - \frac{2}{3}\delta + \frac{5}{12}\right)m^3 + o(m^3)},
\end{aligned}
\tag{17}
$$

and

$$\det(L) = \det(L_{00}) \cdot \det(L_{11}) = N^{\left(-\frac{1}{3}\delta^2 - \frac{1}{3}\delta + \frac{5}{6}\right)m^3 + o(m^3)}.$$

On the other hand, the dimension of the matrix is $n = |\mathbf{I}| = (1 - \delta)m^2 + o(m^2)$. Then we have

$$e^{nm} = N^{(1-\delta)m^3 + o(m^3)}.$$

Therefore as explained in the previous section, we can derive the bound $\delta < 0.292$ by using these values.

5 Computer Experiments

We carry out our computer experiments to check that our lattice construction is valid for recovering short secret key; for various parameters, we compare the solvable key ranges, the determinants and the computational times of lattices between our lattice L and Boneh and Durfee's L_{BD}. The results of our experiments are shown in Table 1 and Table 2. Then we confirm the qualities between two lattice series are equivalent in practice. Moreover, we find the computational time of the L^2 algorithm is reduced by about 30% from the original attack of Boneh and Durfee.

We conduct our computer experiments on the TSUBAME supercomputer [1]. We implement our experiment procedure by the C++ language using Shoup's NTL [12] of version 5.4.2. We carry out the lattice reduction part by the L^2 algorithm [10, 11] with parameter $\delta = 0.99$ and $\eta = 0.51$, [2] and implement the resultant calculation algorithm by [6]. We compile our source code by `gcc-4.1.2` (64 bit version) with `-O6` option.

---

[1]TSUBAME is a grid type supercomputer at Tokyo Inst. of Tech. A node of the supercomputer which we used contains eight Opteron Dual Core model 880 processors of 2.4GHz and 32GB RAM. Note, however, we have not been able to make a parallel version of our algorithm; TSUBAME's massive parallelism has been used only for reducing the total experiment time.

[2]This δ is L^2 algorithm's parameter and different from δ used for defining RSA instance. See the original paper [10, 11] about this δ.

The procedures of our experiments are shown in Figure 2. In step 2-2, The reason for using L^2 algorithm, while we used the LLL algorithm in the analysis, is to speed up the experiment. In fact the L^2 algorithm with NTL is 10 to 50 times faster than our previous implementation of the LLL algorithm (without NTL). Moreover we verify the L^2 algorithm can find sufficiently short vectors for our propose, see Table 2. In step 3-2, we use a parameter $Z = \lfloor\sqrt{X^2Y^2 + 1}\rfloor$ for instantiation, while we used $Z = \sqrt{X^2Y^2 + 1}$ in analysis; this is from our implementation of L^2 algorithm is designed the integer vectors. However we think that this do not affect the quality of the algorithm. In step 2-3 and 3-3, the vectors obtained by the L^2 algorithm are sorted by their length; this is because those vectors are approximate ones and we cannot guarantee that $\mathbf{v}_1$ and $\mathbf{v}_2$ are the shortest two in the reduced basis $\mathbf{b}'_1, \ldots, \mathbf{b}'_n$. In step 2-4 and 3-4, we check the algebraic independence of $h_1(x, y)$ and $h_2(x, y)$ by checking $R(x) \neq 0$ holds or not, where $R(x)$ is the resultant of h_1 and h_2; more precisely, we regard an experiment instance is succeeded if $R(x) \neq 0$ and $R(x_0) = 0$ where x_0 is from the small solution of the target equation.

Input parameters of experiments are ℓ, m and δ which define respectively the bit length of N, the parameter for constructing the lattice, and the ratio of bit length of d to that of N. We carry out the experiments for $m = 6, 8$ and 10, $\ell = 512$ and 1024, and $\delta = 0.260$ to 0.280 in 0.005 intervals.

Results are shown in Table 1 and Table 2. Table 1 shows the solvable key ranges of our lattices and that of [2]; that is, the experiment is succeed or not, for each parameter. The computational times on the table is the average of five experiments for each parameter. Table 2 shows some values for comparing the qualities of L and L_{BD} for some experiment instances; for example, $\log_2$ of the determinants, $\log_2$ of the length of short vectors and other values.

Key recoverability

We check that the difference of the key recoverable ranges between L_{BD} and L. We regard that a lattice can recover the secret key if the polynomials $h_1(x, y)$ and $h_2(x, y)$ pass the check in step 2-4 or 3-4. We carry out our experiments five times for each parameter.

We show the result of the experiments in Table 1. The column "lattice" and "s." mean the type of lattice used in the procedure and number of experiments that pass the check respectively. We also give the "L^2 time", which means the CPU time processing step 2-2 or 3-2, and give the "total time", which means the CPU time processing the subroutine ExpBD or ExpOurs in the table. We conclude that the key recoverable ranges of both algorithms are equivalent in our experiments.

We give some remark on the computational times. We can see the L^2 time is reduced by about 30% compared with the L^2 time of the previous lattice; we think this is caused from the matrix representation of L is simpler than that of L_{BD}. We remark that total computational time is approximately the sum of L^2 time and the time for computing the resultant; thus, we can see the time of the resultant computation is longer than that of L^2 algorithm when m is large. We are able to avoid this by improving the source code for computing the resultant polynomial of bivariate polynomials; however our interest is in the lattices, hence we think that this is not essential for our study.

We further remark on the recoverable range at $\ell = 1024$. We can see the qualities of lattices for $m = 8$ are better than those for $m = 10$ in the table, while we said that the recoverable range expands with larger m. This is caused by an irregular instance; in fact, the fault sample instance for $m = 8$ and $\delta = 0.275$ has the public key $e \approx 2^{1018}$. This is quite smaller than $N \approx 2^{1024}$ while we assumed that $e \approx N$ in our analysis. Since then, this error is caused by that the sample instance is not suitable for our analysis. This shows there are some RSA instances that we may not recover the secret key satisfying Boneh-Durfee bound.

The main procedure of our computer experiments with parameters are m, ℓ and δ.

Step 1: (Make sample RSA instance)　Randomly choose $\ell/2$-bit primes p and q, and let $N = pq$. (In our program, we choose p and q the Euler-Jacobi pseudoprime to bases 2, 3, 5, 7 and 11.)　Randomly choose $\lfloor \delta\ell \rfloor$-bit odd integer as the secret key d such that $\gcd(d, (p-1)(q-1)) = 1$. Compute the public key $e \equiv d^{-1} \pmod{(p-1)(q-1)}$ and let $A = N + 1$, $f_{\mathrm{BD}}(x,y) = -1 + x(A + y)$, $y_0 = p + q$ and $x_0 = (1 - ed)/(p-1)(q-1)$.

Step 2: Execute ExpBD(e,d,N,x_0,y_0; δ,m)

Step 3: Execute ExpOurs(e,d,N,x_0,y_0; δ,m)

The procedure of our computer experiments for L_{BD}.

ExpBD(e,d,N,x_0,y_0; δ,m)

Step 2-1: Let $X = \lfloor N^\delta \rfloor$ and $Y = \lfloor 3N^{0.5} \rfloor$. Compute polynomials $g_{i,j}(x,y)$ in (8) for $I = \{(i,j) \in \mathbb{Z}^2 | 0 \le i \le m, 0 \le j \le 2(1-\delta)i\}$. Then construct the lattice L_{BD} by following [2].

Step 2-2: Apply the L^2 algorithm for L_{BD}.

Step 2-3: Sort the vectors in the reduced basis $\mathbf{b}'_1, \ldots, \mathbf{b}'_n$ by these length to $\mathbf{v}_1, \ldots, \mathbf{v}_n$. Compute $h_c(x,y)$ as the sum of the elements in the deinstantiation of $\mathbf{v}_c$ for $c = 1$ and 2.

Step 2-4: Check $h_1(x_0,y_0) = 0$ and $h_2(x_0,y_0) = 0$. (If $h_1(x_0,y_0) \ne 0$ or $h_2(x_0,y_0) \ne 0$, the experiment is failure.)　Compute $R(x) = \mathrm{Res}(h_1, h_2)$ and check $R(x) \ne 0$ and $R(x_0) = 0$ holds or not.

The procedure of our computer experiments for our L.

ExpOurs(e,d,N,x_0,y_0; δ,m)

Step 3-1: Let $X = \lfloor N^\delta \rfloor$, $Y = \lfloor 3N^{0.5} \rfloor$. Compute polynomials $g_{i,j}(x,y)$ in (8) and convert them to $G_{i,j}(x,y,z)$ for $I = \{(i,j) \in \mathbb{Z}^2 | 0 \le i \le m, 0 \le j \le 2(1-\delta)i\}$. Then construct the lattice L by the method in Section 3

Step 3-2: Apply the L^2 algorithm for L'. Here, L' is the instantiation of L with parameter X, Y (and $Z = \lfloor \sqrt{X^2Y^2 + 1} \rfloor$).

Step 3-3: Sort the vectors of reduced basis $\mathbf{b}'_1, \ldots, \mathbf{b}'_n$ by these length to $\mathbf{v}_1, \ldots, \mathbf{v}_n$. Compute $H_c(x,y,z)$ as the sum of the elements in the deinstantiation of $\mathbf{v}_c$, and $h_c(x,y) = H_c(x,y,-1+xy)$ for $c = 1,2$

Step 3-4: Check $h_1(x_0,y_0) = 0$ and $h_2(x_0,y_0) = 0$. (If $h_1(x_0,y_0) \ne 0$ or $h_2(x_0,y_0) \ne 0$, the experiment is failure.)　Compute $R(x) = \mathrm{Res}(h_1, h_2)$ and check $R(x) \ne 0$ and $R(x_0) = 0$ holds or not.

Figure 2: Our computer experiment procedure for L_{BD} and L

ℓ = 512

Experiment parameters		Lattice	Results		
m	δ		s.	L^{2}time	total time
6	0.265	L_{BD}	5	31.9 sec	49.5 sec
		L	5	22.6 sec	39.9 sec
	0.270	L_{BD}	4	31.0 sec	50.2 sec
		L	4	21.8 sec	40.8 sec
8	0.265	L_{BD}	5	360 sec	721 sec
		L	5	251 sec	610 sec
	0.270	L_{BD}	4	318 sec	613 sec
		L	4	218 sec	514 sec
10	0.270	L_{BD}	5	39 min	159 min
		L	5	28 min	147 min
	0.275	L_{BD}	4	33 min	132 min
		L	4	23 min	121 min

ℓ = 1024

Experiment parameters		Lattice	Results		
m	δ		s.	L^{2}time	total time
6	0.270	L_{BD}	5	123 sec	195 sec
		L	5	86 sec	157 sec
	0.275	L_{BD}	0	112 sec	120 sec
		L	0	75 sec	82 sec
8	0.275	L_{BD}	5	1322 sec	2553 sec
		L	5	860 sec	2081 sec
	0.280	L_{BD}	0	1096 sec	1230 sec
		L	0	695 sec	823 sec
10	0.270	L_{BD}	5	150 min	572 min
		L	5	110 min	575 min
	0.275	L_{BD}	4	127 min	493 min
		L	4	91 min	489 min
	0.280	L_{BD}	0	108 min	267 min
		L	0	78 min	250 min

Table 1: Key recoverability and Computational Time for ℓ=512 and 1024

Determinant and obtained vectors

We compare the determinants, the length of obtained vectors and some amounts to check the qualities of L and L_{BD}; we pick up some instances in our experiments. These are shown in Table 2.

We explain the columns in Table 2. The column deg. shows the degree of the lattice, that is, the number of vectors in the lattice basis. The column D means the value $\log_2(|\mathbf{v}_1|/\det^{1/\deg})$. The column B_1, H_1, B_2 and H_2 respectively mean the value $\log_2|\mathbf{v}_1|$, $\log_2\|h_1(x,y)\|_{XY}$, $\log_2|\mathbf{v}_2|$ and $\log_2\|h_2(x,y)\|_{XY}$.

We give some remarks on the results in the table. The values D in the table are sufficiently smaller than $(\deg - 1)/4$; this is the upper bound guaranteed by Theorem 1 when we use the LLL algorithm. Hence, we verify the L^2 algorithm finds sufficiently short vectors for our objective. We can see values in "result" are equivalent for L and L_{BD}; in fact, we confirmed that they are equivalent at least 10 most significant digits in practice. On the other hand, we have for $c = 1$ and 2, $|\mathbf{v}_c| = \|h_c(x,y)\|_{XY}$ for the lattice L_{BD} and $|\mathbf{v}_c| = \|H_c(x,y,z)\|_{XY} \geq \|h_c(x,y)\|_{XY}/(m+1)$ for the lattice L. That is, the result shows that the inequality is unnecessary pessimistic. In summary, we verified that the obtained polynomials by L are valuable as that by the algorithm of Boneh and Durfee.

6   On the Case of Repetitive Secret Key

In this section we give an application of our simple analysis; we propose a new RSA assumption which we named repetitive secret key.

We assume that the secret key is the repeat of r short bit string d_0 with binary length ℓ. We let $\beta = \log_N d_0$, that is, the rough ratio of the bit-length of N to that of d_0. Hence this is close to $\ell/\log_2 N$.

In this situation we derive the target equation by following the argument in [2]; our equation is

$$f_{\text{rep}}(x,y) = -1 + x(y + A) \pmod{eR} \tag{18}$$

where $R = 1 + 2^\ell + \cdots + 2^{(r-1)\ell}$. We notice that the difference between (1) and (18) is only the

Experiment parameters		RSA Parameters			Lattice		Results					
m	δ	$\log_2 N$	$\log_2 e$	$\log_2 d$	deg.	type	$\log_2 \det$	D	B_1	H_1	B_2	H_1
6	0.265	512.0	511.8	135.0	34	L_{BD}	103989	-2.15	3056.35	3056.35	3056.54	3056.54
						L	103989	-2.15	3056.35	3056.35	3056.54	3056.54
8	0.265	512.0	510.9	135.0	57	L_{BD}	231587	-1.17	4060.96	4060.96	4060.99	4060.99
						L	231587	-1.17	4060.96	4060.96	4060.99	4060.99
10	0.270	512.0	508.2	138.0	86	L_{BD}	436180	1.67	5073.53	5073.53	5073.58	5073.58
						L	436180	1.67	5073.53	5073.53	5073.58	5073.58
6	0.265	1024.0	1022.47	271.0	34	L_{BD}	207700	-5.08	6103.74	6103.74	6103.89	6103.89
						L	207700	-5.08	6103.74	6103.74	6103.89	6103.89
8	0.265	1024.0	1022.65	271.0	57	L_{BD}	463048	-6.74	8116.91	8116.91	8117.11	8117.11
						L	463048	-6.74	8116.91	8116.91	8117.11	8117.11
10	0.270	1024.0	1023.67	276.0	86	L_{BD}	875290	-4.46	10173.33	10173.33	10173.36	10173.36
						L	875290	-4.46	10173.33	10173.33	10173.36	10173.36

Table 2: Determinant and Length of Vectors

modulo. To solve the equation (18), we consider the lattice based attack and we can construct a lattice with lower triangle representation by the technique in this paper.

Hence the determinant of constructed matrix is easily compute by substituting the approximation

$$e \approx N,\ R \approx N^{(r-1)\beta}, X \approx N^{r\beta},\ Y \approx N^{0.5},\ \text{and}\ Z = \sqrt{X^2 Y^2 + 1} \approx N^{r\beta + 0.5},$$

for (17). We further consider the Howgrave-Graham condition $\det(L')^{1/n} < (eR)^m$, where n and m are explained as the outline section.

Finally we obtain the condition for recovering the repetitive secret key as follows.

$$\beta < \frac{r + 3 - \sqrt{r^2 + 6r + 1}}{4}. \tag{19}$$

For $r = 1$, this is equivalent to the result of Boneh and Durfee.

7 Conclusion

In this paper we study the lattice based attack for RSA with short secret key. We give the new simple analysis to obtain their bound $\delta < 0.292$. Through computer experiments, we verify that the recoverable ranges of our lattice and that of Boneh and Durfee's are equivalent, and furthermore, by our approach, we can reduce the computational time of L^2 algorithm by about 30% compared with the one by Boneh and Durfee. One important advantage of our technique is that it does not require any technical method or involved calculation that are necessary in the original technique. We hope that our analysis technique will be applicable in other situations of the lattice based attack.

Acknowledgement

References

[1] Y. Aono, A new lattice construction for partial key exposure attack for RSA, in *Proceedings of PKC 2009*, Lecture Notes in Computer Science, vol. 5443, pp. 34-53, 2009.

[2] D. Boneh and G. Durfee, Cryptanalysis of RSA with private key d less than $N^{0.292}$, *IEEE Transactions on Information Theory*, vol. 46, No. 4, pp. 1339-1349, 2000.

[3] J. Blömer and A. May, Low Secret Exponent RSA Revisited in *CaLC 2001*, Lecture Notes in Computer Science, vol. 2146, pp. 4-19, 2001.

[4] D. Coppersmith, Finding a small root of a univariate modular equation, in *Proceedings of EUROCRYPT* 1996, Lecture Notes in Computer Science, vol. 1070, pp. 155-165, 1996.

[5] M. Ernst, E. Jochemsz, A. May, and B. Weger Partial key exposure attacks on RSA up to full size exponents, in *Proc. of EUROCRYPT'05*, Lecture Notes in Computer Science, vol. 3494, pp. 371-386, 2005.

[6] A. D. Healy, Resultants, Resolvents and the Computation of Galois Groups, Available online at `http://www.alexhealy.net/papers/math250a.pdf`.

[7] N. Howgrave-Graham, Finding small roots of univariate modular equations revisited, in *Proceedings of Cryptography and Coding*, Lecture Notes in Computer Science, vol. 1355, pp. 131-142, 1997.

[8] E. Jochemz and A. May, A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants, in *Advances in Cryptology (Asiacrypt 2006)* Lecture Notes in Computer Science, vol. 4284, pp. 267-282, 2006.

[9] A. K. Lenstra, H. W. Lenstra Jr. and L. Lovász Factoring polynomials with rational coefficients, *Mathematische Annalen*, vol. 261, No. 4, pp. 515-534, 1982.

[10] P. Nguyen and D. Stehlé, Floating-Point LLL revisited, in *Proceedings of EUROCRYPT 2005*, Lecture Notes in Computer Science, vol. 3494, pp. 215-233, 2006.

[11] P. Nguyen and D. Stehlé, Floating-Point LLL (Full version), available online at `ftp://ftp.di.ens.fr/pub/users/pnguyen/FullL2.pdf`.

[12] V. Shoup, NTL: A Library for doing Number Theory, available online at `http://www.shoup.net/ntl/index.html`.