Coppersmith's lattices and "focus groups": an attack on small-exponent RSA

Stephen D. Miller[*], Bhargav Narayanan[†],
{miller,narayanan}@math.rutgers.edu
and Ramarathnam Venkatesan
venkie@microsoft.com

December 16, 2020

Abstract

We present a principled technique for reducing the lattice and matrix size in some applications of Coppersmith's lattice method for finding roots of modular polynomial equations. It relies on extrapolating patterns from the actual behavior of Coppersmith's attack for smaller parameter sizes, which can be thought of as "focus group" testing. When applied to the small-exponent RSA problem, our technique reduces lattice dimensions and consequently running times, and hence can be applied to a wider range of exponents. Moreover, in many difficult examples our attack is not only faster but also more successful in recovering the RSA secret key. We include a discussion of subtleties concerning whether or not existing metrics (such as enabling condition bounds) are decisive in predicting the true efficacy of attacks based on Coppersmith's method. Finally, indications are given which suggest certain lattice basis reduction algorithms (such as Nguyen-Stehlé's L2) may be particularly well-suited for Coppersmith's method.

Keywords: Factoring, small exponent RSA, lattice attacks, lattice basis reduction, Coppersmith's method

[*]Supported by NSF grants CNS-1526333 and CNS-1815562.
[†]Supported by NSF grant DMS-1800521.

1 Introduction

Ever since Shamir's devastating attack on the Knapsack cryptosystem [19], lattice basis reduction algorithms such as LLL [15] have found surprising success against cryptosystems that *a priori* have nothing to do with lattices. A fundamental example is the RSA cryptosystem [18], whose public key consists of an integer $n = pq$ (where p and q are large secret primes of comparable size) and an encryption exponent e. In situations where some extra information about the public key is known (e.g., certain bits of p or q), it is sometimes possible to use the lattice basis reduction method of Coppersmith [7] to discover the factorization of n.

One notable such situation is when the secret decryption exponent d is *small*,

$$d \;=\; n^{\delta}\,, \quad \delta < \tfrac{1}{2} \tag{1.1}$$

(see Section 2 for more background on RSA). Wiener [21] showed that continued fractions expose d when $\delta < 1/4$, essentially instantaneously. Continued fraction approximations can be thought of as the simplest example of lattice basis reduction, namely for 2-dimensional lattices. Boneh-Durfee [4] apply Coppersmith's method with higher dimensional lattices to give an attack for

$$\delta \;<\; 1 \,-\, 2^{-1/2} \;\approx\; .292\,. \tag{1.2}$$

More precisely, they prove that LLL's output on a particular lattice produces enough information to factor n (subject to an algebraic independence assumption). It is an important open problem to improve[1] the bound (1.2), which still stands as the current record despite many attempts to improve it, or to even rigorously establish the needed algebraic independence (see [2]).

[1]This raises the question of what precisely constitutes an improvement over Boneh-Durfee's .292 result [4]. Merely replacing .292 by a larger constant alone is insufficient, since algebraic dependence may creep for smaller δ. Indeed, Bauer [1] gives an attack in which the right-hand side of (1.2) can be replaced by .34, but which suffers from a failure of algebraic independence at a much earlier point. On the other hand, it should be pointed out that we don't even know at present whether or not Boneh-Durfee's attack suffers from the same problem before reaching $\delta = .292$. See the end of Section 2 for more comments.

On the practical side, one sees another obstacle from (1.3): attacks which require enormous lattices are infeasible and cannot come close to their theoretical limitations anyhow. Our approach here is to develop a method which gives experimentally verified improvements over previous work.

Since the LLL algorithm has a widespread reputation for outperforming its provable guarantees, one might surmise that the bound (1.2) is more modest than actual experiments would indicate. Surprisingly, the opposite is true: all successful experiments in the literature work only for δ relatively far below the theoretical upper bound of $1-2^{-1/2} \approx .292$ [4, 5, 22]. The reason for this is that (1.2) is an asymptotic estimate that requires very large lattices. Specifically, for 1,024-bit moduli n

$$\begin{array}{c} \text{Boneh-Durfee's attack requires lattices} \\ \text{of dimension} \geq 500 \text{ for } \delta \geq .278. \end{array} \tag{1.3}$$

The difficulty of finding short vectors in lattices of dimension ≥ 500 already serves as the hard underlying problem behind other cryptosystems, and indeed $\delta = .278$ is close to the limit of known experiments in [4, 5, 22].[2]

Thus implementing lattice-based attacks can itself face impractically difficult problems even in ranges covered by theoretical guarantees. It is therefore natural to ask the following questions:

Q1. Can one practically solve small-exponent RSA instances for δ significantly larger than the experiments reported in [4, 5, 22]?

Q2. Is there a barrier from algebraic independence that creeps in before the theoretical upper bound is reached? If so, how does one estimate the true range of validity of the attack?

Q3. How can Coppersmith's method be modified to reduce the size of the matrices (and their entries) involved? It has long been considered to look at sublattices, but what are the optimal sublattices to choose?

In Section 2 we review Coppersmith's method and the Boneh-Durfee attack, and comment on some nuances of comparing theoretical analyses to actual outcomes in applications of Coppersmith's method (e.g., **Q2**). The main contribution of this paper is to **Q3**, by introducing a method in Section 3 (influenced by the idea from machine learning of trying to find patterns in known examples, rather than being guided solely by theory) to cut down the matrix size and hence push back the choke point that high dimensional

[2] It follows that it is important to distinguish between theoretical ranges of applicability of Coppersmith's method, and the practical ranges that they could possibly be applied to (in light of the difficulty of lattice basis reduction in large dimensions).

lattice basis reduction algorithms face in practice. We use that to address **Q1** in Section 4, where we show our method is faster (and often more effective).

In Section 5 we present another example of how the "focus group" method can be applied to the attack in [3, §6], where the size of the spanning set can be reduced significantly. We chose these two examples because they are prominent theoretical and practical applications of Coppersmith's method to RSA. All the computations here (unless otherwise noted) were performed in Mathematica[3] v.11 on a Dell PowerEdge R740xd server equipped with two Intel Xeon Silver 4114 2.2GHz processors and 256GB RAM. In particular we did not use specialized lattice basis reduction packages such as [9, 20].

We would like to thank Dan Boneh, Henry Cohn, Nadia Heninger, Jeff Hoffstein, Antoine Joux, Daniel Lichtblau, Alexander May, Oded Regev, Adi Shamir, Noah Stephens-Davidowitz, and David Wong for their helpful discussions. We are also very appreciative of the anonymous referee for their very helpful comments, and to Galen Collier and the staff of the Rutgers Office of Advanced Research Computing for their assistance with Rutgers' Amarel high performance cluster.

2 An overview of Coppersmith's method and Boneh-Durfee's attack on RSA

As before, let p and q be secret large prime numbers of comparable size, and $n = pq$ the public RSA modulus. Let e be the public encryption exponent and $d = n^\delta$ be the secret decryption exponent, which satisfy $ed \equiv 1 \pmod{\phi(n)}$, where $\phi(n) = (p-1)(q-1) = n - p - q + 1$. In this case d's relation to e can be restated as the existence of an integer k such that

$$e\,d \;=\; 1 \;+\; k\,\phi(n)\,, \quad \text{where } k \;\approx\; n^\delta\,, \tag{2.1}$$

in which we have made the natural – and trivially verifiable – assumption that the public exponent e has comparable size to n. After dividing both

[3]Specifically, using Mathematica's `LatticeReduce` command, which implements the L2 algorithm of Nguyen and Stehlé [17].

sides by $d\phi(n)$ and using the fact that $n - \phi(n) = O(\sqrt{n})$, this implies

$$\begin{aligned} \left|\frac{e}{n} - \frac{k}{d}\right| &\leq \left|\frac{e}{n} - \frac{e}{\phi(n)}\right| + \left|\frac{e}{\phi(n)} - \frac{k}{d}\right| \\ &= O\left(\frac{e}{n^{3/2}}\right) + \frac{1}{|d\phi(n)|} = O(n^{-1/2})\,. \end{aligned} \tag{2.2}$$

Wiener [21] observed that if $\delta < \frac{1}{4}$, the fraction $\frac{k}{d}$ approximates $\frac{e}{n}$ much more accurately than $d^{-2} \gg n^{-2\delta}$, which is an unusually good approximation of a real number by a rational number of denominator d. Hence $\frac{k}{d}$ occurs among the continued fraction approximants to $\frac{e}{n}$, and can be very efficiently computed.

Following [4], consider the bivariate polynomial

$$f(x,y) = x(n-y) + 1\,, \tag{2.3}$$

which according to (2.1) satisfies

$$f(x_0, y_0) \equiv 0 \pmod{e}\,, \tag{2.4}$$

where

$$x_0 = k = O(e^{\delta}) \quad \text{and} \quad y_0 = n - \phi(n) = O(\sqrt{e}) \tag{2.5}$$

are both relatively small compared to the modulus e. Coppersmith's method (in this example, following Howgrave-Graham [11]) is used to promote a polynomial congruence relation such as (2.4) into a system of two integer polynomial equalities, which can then be solved using classical methods. To illustrate this in terms of the Boneh-Durfee attack, define x-shifts and y-shifts

$$\begin{aligned} g_{i,\ell,m}(x,y) &= x^i\, f(x,y)^\ell\, e^{m-\ell} \\ \text{and} \quad h_{j,\ell,m}(x,y) &= y^j\, f(x,y)^\ell\, e^{m-\ell}\,, \end{aligned} \tag{2.6}$$

for

$$0 \leq \ell \leq m\,,\ 0 \leq i \leq m - \ell\,, \quad \text{and} \quad 1 \leq j \leq t\,. \tag{2.7}$$

They satisfy

$$g_{i,\ell,m}(x_0, y_0) = h_{i,\ell,m}(x_0, y_0) \equiv 0 \pmod{e^m} \tag{2.8}$$

and span a sublattice Λ of $\mathbb{R}[x, y]$, the latter of which is endowed with the sum-of-squares norm $\|\cdot\|$ on polynomial coefficients. A short vector in this sublattice is a polynomial with small coefficients, ideally small enough that its value at a particular point such as (x_0, y_0) will itself be relatively small. By (2.8), that value is also a multiple of e^m; thus if it is less than e^m in absolute value, it must actually vanish.

To make this more precise in our setting, let X and Y be upper bounds for $|x_0|$ and $|y_0|$, respectively (such as provided in (2.5)). Howgrave-Graham [11] observed that if a polynomial $p(x, y) \in \Lambda$ satisfies[4]

$$\|p(xX, yY)\| \quad < \quad \frac{e^m}{\sqrt{w_p}}\,, \tag{2.9}$$

where w_p is the number of nonzero monomials in $p(\cdot, \cdot)$, then an application of Cauchy-Schwartz shows $|p(x_0, y_0)| < e^m$. In particular, (x_0, y_0) is a root of $p(\cdot, \cdot)$ over $\mathbb{Z}$ since $p(x_0, y_0) \equiv 0 \pmod{e^m}$. Boneh-Durfee prove that this norm condition is met for the shortest vector outputted by LLL provided

$$|\Lambda| \quad \leq \quad e^{m(w-1)}(w2^w)^{(1-w)/2}\,, \quad w \,=\, \dim(\Lambda)\,, \tag{2.10}$$

where $|\Lambda|$ denotes the covolume of Λ. For $\delta < \frac{7}{6} - \frac{\sqrt{7}}{3} \approx .284$ this "enabling" condition is met for sufficiently large values of m and e. We shall refer to this as the Boneh-Durfee ".284" attack, in order to distinguish it from their more refined analysis (using a carefully selected sublattice) that extends the range to $\delta < 1 - 2^{-1/2} \approx .292$. See also [5, 12, 14] for other attacks theoretically deriving exponents of this size, or close to it.

Under the enabling condition (2.10), the two shortest vectors outputted by LLL are coefficients of bivariate polynomials which vanish at (x_0, y_0). Boneh-Durfee point out that in practice these polynomials are algebraically independent, and thus their common roots can be extracted using resultants. However, this observed algebraic independence has not yet been rigorously established (see [2]). In principle (as can happen, for example, if one does not use any of the polynomials $h_{j,\ell,m}$) the shortest vectors may all result in

[4]Note that vector length $\|\cdot\|$ is not necessarily the appropriate metric. The length condition (2.9) is quadratic in the polynomial coefficients, but the actual value of interest (the polynomial evaluated at a particular point) is instead linear: it is the value of a linear functional on Λ. Of course bounding the norm bounds the value of a linear functional, but possibly with a significant loss. One might imagine leveraging some geometric information about x_0 and y_0 (such as their sign) which is known in advance.

polynomials which are trivial multiples of each other, and hence not give enough equations to reveal the two unknowns x_0 and y_0.

An interesting example: consider the 1,000 bit RSA modulus $n = pq$ and key $d = n^\delta$ given by

$$p = 327534248375076317083641611376534056264358811260976111454743\\ 469579874653650577266211366585026890270802159105074832098421\\ 51169272587144341747240549531 33\,,$$

$$q = 327462704072360233831723075103626846066746692190298143145154\\ 087005180715732984190358817594057449905589163120424047417288\\ 340023937447137939357162457765 7\,,$$

and

$$d = 300147077152565471186517713474704374146330287118250537992743\\ 5326735028048350149451\,, \quad (\delta \approx .2707).$$

We applied the BKZ lattice basis reduction from [20] with block size 3 to the lattice from Boneh-Durfee's .292 attack with parameters $(m, t) = (5, 2)$. Of the 25 output vectors, only one of them (the fifth longest[5]!) produces a polynomial which vanishes at (x_0, y_0). Interestingly and perhaps counterintuitively, applying BKZ with larger block size (such as 5) failed to produce *any* vectors vanishing at (x_0, y_0). (We were unable to do any better using the implementation of BKZ in `sagemath`.) This example demonstrates that the vector length of the output basis is not the sole determinant of success, in addition to the sensitivity to the choice of lattice basis reduction method.

It is worth mentioning other lattice attacks which use polynomials different from (2.3), and which also consistently beat Wiener's $\delta < 1/4$ bound. Bauer's thesis [1, Chapter 4] discusses a three-variable analog based on using a short continued fraction approximation of e/n, stopping roughly at the point at which it is theoretically expected to differ from that of $e/\phi(n)$. Two additional integer parameters are then substituted to account for the remain-

[5] A similar feature was already observed in A. Bauer's Ph.D. thesis [1].

ing part of the continued fraction approximation.[6] A lattice is again formed as above using congruences modulo powers of e. Her analysis of the enabling condition shows that when $\delta < .34$, the lattice has short vectors that produce polynomials which vanish at the desired roots. Alternatively, one can instead apply the lattice method of [13] to this partial continued fraction approach (as we have attempted in experiments) – its analogous enabling condition holds for $\delta < 1/3$. These are theoretical ranges in which Coppersmith's method will provably find short vectors (for large enough lattice sizes), yet possibly nevertheless fail to factor n because of algebraic dependence. Both ranges extend much further than Boneh-Durfee's $\delta < 1 - 2^{-1/2} \approx .292$ range, and the lattice sizes in both attacks can be improved using the "focus group" methodology in Section 3 below.

Indeed, despite the promising increase in this range for δ from .292 to 1/3 or .34, neither of these approaches comes near .292 in practice. The results of our limited experimental trials indicate that the actual performance of either of these algorithms seems roughly comparable to that of Boneh-Durfee's. In particular, the experiments show that algebraic independence fails at a much earlier point, well before the enabling condition of $\delta < 1/3$ or $\delta < .34$ is reached. Furthermore, the lattice sizes necessary to study such large exponents δ are themselves impractically large. That calls into question the direct practical relevance of the enabling condition itself, and demonstrates the importance of a better understanding of the actual performance of these attacks.

Remarks on sublattices and minimizing $|\Lambda|$: in order to leverage provable guarantees that a lattice basis reduction algorithm will find a sufficiently short vector, sublattices in variants of Coppersmith's attack are often taken in order to effectively reduce the covolume $|\Lambda|$ (see **Q3** in Section 1). This is primarily done to ensure the validity of an enabling condition such as (2.10). While this allows for rigorous, theoretical analysis, there are geometric reasons why it may not be optimal:

- If Λ does not behave like a random lattice, it may have vectors at several length scales that do not interact much with each other.

[6] See also the paper [8]. The anonymous referee of the present paper kindly suggested another approach: attempt to guess a portion of the bits of these unknown parameters, in the hopes of gaining a more-than-compensating savings from the smaller lattices which result from analyzing the remaining portion.

- For example, suppose one appends a very long vector to a short basis of a lattice perpendicular to it. This would magnify the covolume without affecting the outcome of lattice basis reduction at all.

- The ultimate goal in Coppersmith's method is not to reduce the covolume, but to increase the likelihood of finding a short vector. It is more important to identify sublattices having short vectors, which is not well-measured by the covolume.

- As we have noted in the above discussion of Bauer's thesis [1], attacks with very different enabling condition bounds may perform similarly in practice, since algebraic dependence may creep in before the enabling condition is reached. Thus $|\Lambda|$ itself may not actually enter into a meaningful bound anyhow.

We conclude this section by remarking that the lattices produced in the Boneh-Durfee attack appear to be far from random, as is evidenced by their vector lengths. This appears to be in contrast with the lattices produced in other applications of Coppersmith's method – though not all (e.g., [6]). For example, one typically expects a basis outputted by the LLL algorithm [15] to have vectors of comparable length. Figure 1 shows the logarithms of the vector lengths in the original and reduced lattice bases for an instance of the Boneh-Durfee .284 attack with $n \approx 2^{6,000}$ and $\delta \approx .284$. At this logarithmic scale one can see clumps of basis vectors of roughly the same length, yet nevertheless the overall lengths of the basis vectors do differ significantly within each plot. The plot the left indicates that the input basis has several different regimes, owing to the structure of (2.6)-(2.7). The plot on the right shows that the output basis also has vectors in (fewer) clumps of similar logarithmic length, in particular with a large separation between the shortest vector (which represents a constant polynomial) and the others. Not surprisingly, the attack failed in this particular instance. Understanding this "clumping" phenomenon may help gain insight into Coppersmith's method. For example, is there inhomogeneity in the geometry of the lattice that effectively reduces its dimension? If so, can it be exploited? This is the underlying geometric motivation behind the "focus group" method in Section 3, which is an approach to identifying sublattices having short vectors.

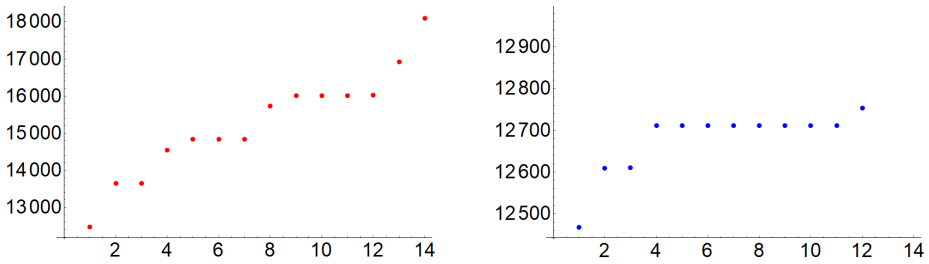


Figure 1: Natural logarithms of lengths of lattice basis vectors in the Boneh-Durfee .284 attack with $n \approx 2^{6{,}000}$ and $\delta \approx .284$, before and after lattice basis reduction. The attack failed in this instance.

3 "Focus group" attacks

Applying lattice basis reduction to a sublattice may increase the chances of finding short vectors, while of course simultaneously decreasing run time. Given the inherent limitations in performing lattice basis reduction in high dimensions, sometimes finding an appropriate sublattice can make the difference between finding solutions and finding no solutions at all (or not even being able to fully execute lattice reduction).

In this section we describe a principled, evidence-based approach to selecting a sublattice in certain lattice basis reduction problems, such as applications of Coppersmith's method. Its main idea is to deform to a simpler problem in which one can directly determine which basis vectors contribute nontrivially to the shortest vectors. This methodology is applied in Section 4 to small-exponent RSA and in Section 5 to the "Coppersmith in the wild" smart card attack of [3].

This "focus group" attack consists of three main steps:

1. **Set small parameters.** Find a regime with the same lattice dimension, but reduced sizes of basis vectors coordinate entries. This makes it faster (or even possible) to execute lattice basis reduction on large matrices. For example, in the case of small-exponent RSA we set δ in (1.1) to be slightly larger than $\frac{1}{4}$ (which is the point at which Wiener's continued fraction attack ceases to work). Of course would be desirable to improve this step by theoretically understanding in advance which basis vectors to keep (e.g., using the notion of "helpful vector" from [16, Chapter 7] rather to rely on experiments), but this may not be practical because of the complexity of lattice basis reduction.

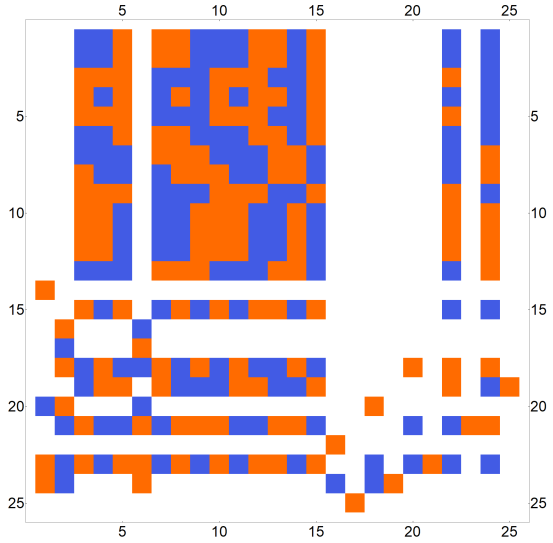


Figure 2: A representation of the change of basis matrix for the lattice basis reduction step in Boneh-Durfee's .284 attack (see the text for more details). The matrix has a number of columns with many zero entries (marked white).

2. **Check the output to see which parts of the original basis were actually used.** Figure 2 pictorially represents the change of basis matrix for the lattice basis reduction step in Boneh-Durfee's .284 attack for a 6,000-bit RSA modulus n, with $\delta \approx .251$ and parameters $(m, t) = (4, 2)$ (see (2.7)). The columns are indexed by the input basis vectors and the rows are indexed by the output basis vectors. Each entry in the matrix is plotted as blue/dark gray (positive), orange/light gray (negative), or white (zero).

   The long white vertical streaks emanating from the top of the figure reveal that certain input basis vectors are not used in forming the shortest vectors in the lattice output. Those basis elements from (2.6)-(2.7) can be graphically represented as in Figure 3, where the figure on the left represents the x-shifts and the figure on the right represents the y-shifts. Here the unfilled white circles indicate unused vectors and filled black circles indicate useful vectors. Boneh-Durfee's .292 attack refines their .284 attack by discarding some y-shifts from (2.7), but not the same ones as here (theirs are chosen to minimize the covolume $|\Lambda|$).

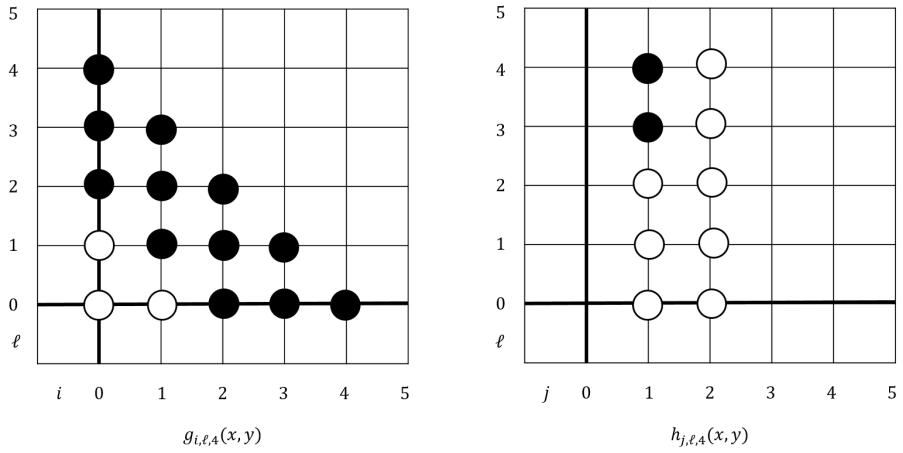


Figure 3: A representation of which polynomials in (2.6)-(2.7) are actually used (black circles) in forming the shortest vector in the lattice basis reduction step for a particular instance of the Boneh-Durfee attack. The unfilled, white circles represent discarded basis vectors (see the text for more details).

In fact, the figure indicates that most of the y-shifts are not actually used. It is more striking that some of the smaller x-shifts are not used, consistent with the utility of a similar device in [5, §4]. Similar patterns arise for larger parameter sizes and were used to formulate the attack in Section 4. Indeed, examples of patterns yield useful descriptions in terms of extra additional parameters, which are then used to extrapolate good guesses for which families of sublattices to look at in more challenging situations.

3. **Remove unused basis elements.** This has advantages for run time, storage, and quality of results, since lattice basis reduction on smaller lattices typically performs dramatically better. After all, the approximation factor in lattice basis reduction is tighter in smaller dimensions.

4 The "focus group" attack on small-exponent RSA

We now specialize the methodology of Section 3 to small-exponent RSA. Trials of the Boneh-Durfee .284 attack [4] with small parameters suggest a particular sublattice to use, which we shall describe below. Previous work has selected sublattices using other methods. For example, Boneh-Durfee suggest in their .292 attack to remove certain $h_{j,\ell,m}$ which contribute large factors to the covolume. Later work by Blömer and May [5] suggests removing some of the $g_{i,\ell,m}$ as well (see also [12, 14]).

Our approach is guided by which vectors are likely to contribute to a nontrivial solution, but not directly by determinant considerations. We introduce two integer parameters σ and τ (in addition to m and t), and exclude from (2.7) all indices with $i + \ell \leq \sigma$ and $\ell - 2j \leq \tau$ (this is motivated by the shape of the black and white circles in Figure 3). That is, the polynomials in (2.6) are taken for indices

$$(4.1)\quad 0 \leq \ell \leq m,\ \max(-1, \sigma - \ell) < i \leq m - \ell,\ \text{ and } 1 \leq j \leq \min(t, 1 + \tfrac{\ell - \tau}{2})$$

instead of (2.7). We choose $X = \lceil 2e^{\delta} \rceil$ and $Y = \lceil 2e^{1/2} \rceil$ as rough integral upper bounds for x_0 and y_0, respectively (cf. (2.5)).

Experiments

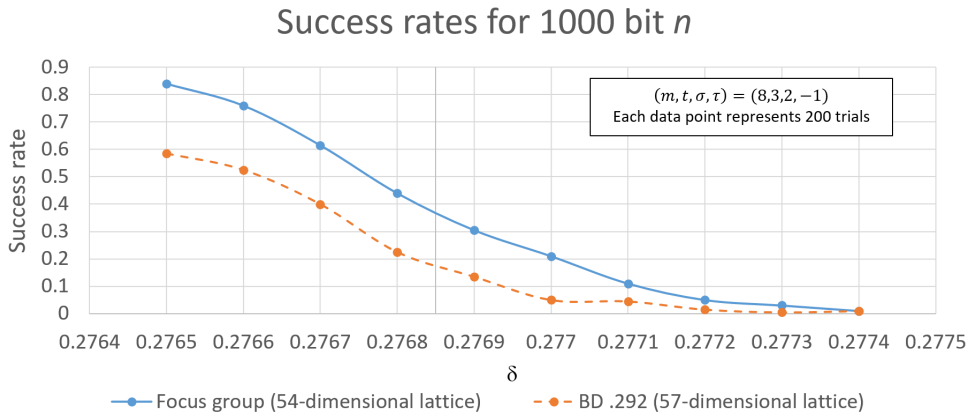
We ran timings using Mathematica v.11 on a Dell PowerEdge R740xd server with two Intel Xeon Silver 4114 2.2GHz processors and 256GB RAM. We did not seriously attempt to optimize the lattice basis reduction computations, relying instead on Mathematica's `LatticeReduce` command (which is an implementation of [17]). Timing results are presented in Table 1 and include, as a control experiment, a comparison with an implementation of Boneh-Durfee's .292 and .284 attacks using Mathematica on the same machine with the same parameters m and t. It would be interesting to perform a similar comparison with the algorithm of [5], whose sublattice is more similar to the one selected by the "focus group" attack. The attack in [5] also satisfies the enabling condition for the same $\delta < 1 - \sqrt{1/2} \approx .292$ range as Boneh-Durfee's attack [4]. We have not rigorously analyzed at what point our enabling condition breaks down, as it may be moot anyhow: algebraic independence might be lost before that point (see the comments at the end of Section 2).

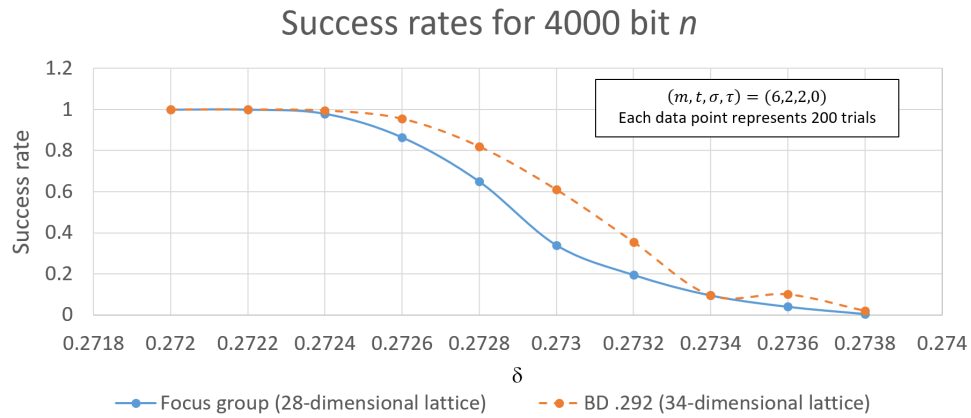
bits of n	trials	δ	(m,t,σ,τ)	Focus group		Boneh-Durfee .292		Boneh-Durfee .284	
				time	dim	time	dim	time	dim
1000	100	.270	(6,2,2,0)	8.46 s	28	15.95 s	34	30.29 s	42
4000	100	.273	(6,2,2,0)	60.37 s	28	103.49 s	34	182.92 s	42
6000	15	.277	(8,3,2,-1)	1905.81 s	54	2170.15 s	57	3920.45 s	72
10000	100	.260	(3,1,1,0)	1.70 s	14	2.76 s	17	4.47 s	20
10000	100	.265	(4,1,1,0)	12.94 s	14	17.90 s	17	24.76 s	20
10000	15	.277	(8,3,2,-1)	4565.47 s	54	5063.59 s	57	8492.21 s	72

Table 1: Timings of trials of the "focus group" attack on small-exponent RSA. Times listed are averages over many trials of RSA keys, in which each of the three attacks on the right is performed on the same key. Comparisons are given in the last two columns for the Boneh-Durfee .292 and .284 attacks with the same values of m and t. We list the average time in seconds as well as the dimension of the lattices involved, which get progressively larger as one goes from the focus group attack to the Boneh-Durfee .284 attack. The focus group attack is significantly faster, and uses less memory due to the smaller lattice size. (All computations were run on the same machine using the same lattice basis reduction algorithm.) Times refer to the lattice basis reduction step only.

bits of n	trials	δ	(m,t,σ,τ)	Focus group		Boneh-Durfee .292		Boneh-Durfee .284	
				Success %	dim	Success %	dim	Success %	dim
1000	100	.270	(6,2,2,0)	100%	28	100%	34	100%	42
1000	100	.273	(6,2,2,0)	43%	28	64%	34	64%	42
1000	100	.277	(8,3,2,-1)	21%	54	10%	57	9%	72
1000	100	.279	(10,4,2,-3)	62%	92	26%	85	26%	110
1000	100	.280	(10,4,2,-3)	1%	92	0%	85	0%	110
4000	100	.273	(6,2,2,0)	58%	28	100%	34	100%	42
6000	15	.277	(8,3,2,-1)	100%	54	100%	57	100%	72
10000	100	.260	(3,1,1,0)	100%	14	100%	17	100%	20
10000	100	.265	(4,1,1,0)	100%	14	100%	17	100%	20
10000	15	.277	(8,3,2,-1)	100%	54	100%	57	100%	72

Table 2: Success rates of the three algorithms in Table 1, as measured by producing multiple polynomials which vanish on the secret key (as a time-saving proxy to allow for more experiments to be run). Again, each trial involves the three attacks on the right applied to the same RSA key. The focus group attack has a lower success rate than Boneh-Durfee's attacks for some smaller values of δ, but appears to outperform the Boneh-Durfee attack for larger values of δ and lattice dimensions. More data is shown in the plots in Figure 4.





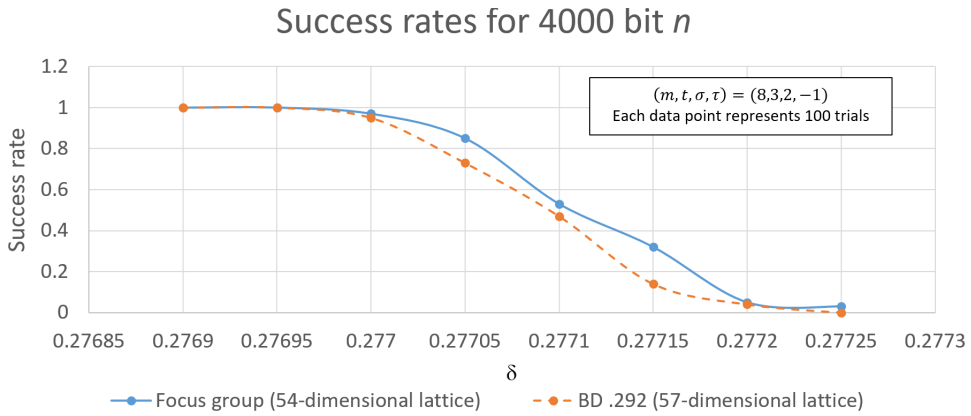


Figure 4: These plots amplify the data in Table 2 and show how the focus group method compares to the Boneh-Durfee .292 attack near their limits of failure for larger exponents δ. It appears the focus group method outperforms the Boneh-Durfee .292 attack for larger δ and lattice dimensions.

Extraction of the secret key after lattice basis reduction can be fairly slow, but we always found it to be possible so long as more than one output polynomial vanishes on the secret key. In order to generate more data on the success of the method and avoid this bottleneck, we ran a number of experiments in which success was measured by finding more than one such polynomial. (In order to be more confident that no algebraic dependence issues arose, we performed a more rigorous analysis on several sample lattice reduction outputs and found sufficient algebraic independence in all cases.) Table 2 and Figure 4 show the probability of success for the focus group method compared to the Boneh-Durfee attacks. These experiments were run on many machines in Rutgers' Amarel high performance cluster. For some parameter choices (m, t, σ, τ), the focus group attack is more successful than the Boneh-Durfee attacks (e.g., in the first plot in Figure 4), but not for all. For example, in the entry for 4000 bit moduli n with $(m, t, \sigma, \tau) = (6, 2, 2, 0)$ and $\delta = .273$, the Boneh-Durfee attack was successful on 100 out of 100 randomly chosen RSA keys, whereas the focus group attack was successful only 58 of those same 100 RSA keys. The second plot in Figure 4 shows this holds true for the same choice of $(m, t, \sigma, \tau) = (6, 2, 2, 0)$ and for exponents δ in a nearby range. However, it appears this phenomenon may be limited to smaller δ and matrix sizes: when one instead considers the larger lattices with $(m, t, \sigma, \tau) = (8, 3, 2, -1)$ (the last plot in Figure 4), the situation apparently reverses and the focus group method is more successful than Boneh-Durfee's attack. Table 2's entries for 1000 bit n and the first plot in Figure 4 also suggest that the focus group may be more successful than Boneh-Durfee's attack in runs that involve larger lattices and exponents δ. We should caution, however, that we have not systematically analyzed this beyond the experiments presented here. We are also unsure exactly what to attribute this apparent improvement to, though we suspect it is that lattice basis reduction algorithms have superior performance on smaller lattices.

As an aside, the timings demonstrate the power of the implementation of the L2 lattice basis reduction algorithm [17] used in Mathematica's `LatticeReduce` command, which typically outperformed the BKZ implementations in NTL and `sagemath`. For example, the exponents δ achieved here are higher than in previously reported experiments (e.g., [4, 5, 22]), with the size of d in the last entry in Table 1 being 220 bits longer than achieved in [4] for a 10,000-bit RSA modulus n. Recall that Table 1 makes a controlled comparison between different approaches to Coppersmith's method on the same RSA keys, by using the same hardware and same lattice basis reduction al-

gorithms. It is interesting to speculate whether certain features of [17] are particularly useful when applied to the lattices produced by Coppersmith's method, and if so, how to leverage them further (recall also the example in Section 2, where BKZ is more helpful with block size 3 than block size 5).

5 The "focus group" attack applied to partial-key-recovery methods of [3]

In this section we apply the "focus group" methodology to the "Coppersmith in the wild" partial-key-recovery attack of [3, §6], which as far as we are aware was the first Coppersmith-style attack successfully applied to real-world keys. In their situation, $n = pq \approx 2^{1024}$ is an RSA modulus, and $p \approx 2^{512}$ has the special form

$$p \;=\; a \;+\; 2^t s \;+\; r\,, \tag{5.1}$$

where a and t are – or at least suspected to be – known, but s and t are unknown. The authors consider the polynomial $f(x, y) = a + 2^t x + y$, and form the $\binom{k+2}{2}$-dimensional lattice spanned by the polynomials

$$\{x^i y^h f(x, y) | 0 \le i + h \le k - 1\} \;\cup\; \{x^j y^h n | 0 \le j + h \le k\} \tag{5.2}$$

for an integer parameter $k \ge 0$. Lattice basis reduction is then used (if successful) to obtain two small algebraically independent polynomials, from which the values of s and t are extracted.

We performed experiments with one particular set of parameters successfully studied in [3, §6.2], namely $k = 4$, $t = 428$, $a = 2^{511} + 2^{510}$, $X = 2^{100}$, and $Y = 2^{28}$. Experiments with small parameters revealed that solutions are often found within sublattices generated by the polynomials in (5.2) having $h \ge h_0$, where h_0 is fixed (see Figure 5).[7] Following the focus group methodology, we considered such a sublattice in this particular example with $h_0 = 1$, and verified that by taking these ten polynomials (instead of the 25 in (5.2)), we could just as easily recover the factorization of n using lattice basis reduction applied to the corresponding sublattice. Thus the same performance

[7] After removing a common factor of y^{h_0}, the generators of this sublattice are given by (5.2) but with k replaced by $k - h_0$. This might explain the remark on [3, p. 355] that some experiments for small k worked in situations where theoretically no useful output was expected.

in this example is obtained from using this chosen ten-dimensional sublattice of the ambient 25-dimensional lattice.

The use of Hermite normal form in the "focus group" method: Hermite normal form has previously been applied to Coppersmith's method (e.g., [10, Example 19.3.2]). For example, one can obtain the Hermite normal form of matrix whose rows form a generating set of the lattice (perhaps strictly larger than a basis), and then apply lattice basis reduction to only some of the rows of the output. Thus instead of using the generating set directly produced by Coppersmith's method, we instead transform it to a different generating set (using Hermite normal form) and apply the focus group methodology of selecting a sublattice at this secondary stage instead.

This approach via Hermite normal form did not work well when applied to the previous example, but did find success when applied to primes (5.1) of a different type than those studied in [3]. Namely, here we arbitrarily chose a to be the smallest positive integer congruent to $3^{1,000} \pmod{2^{512}}$, and randomly generated p of the form (5.1) with $X = Y = 2^{25}$, and $t = 488$. We took $k = 5$ and studied the 36×21 matrix of coefficients of the generating set (5.2) with respect to the monomial basis $\{x^i y^h | 0 \leq i+h \leq k\}$, and considered the lattice spanned by the top $\binom{k+1}{2} = 15$ rows of its Hermite normal form as generated by Mathematica's `HermiteDecomposition` command. Lattice basis reduction applied to this 15-dimensional lattice then yielded polynomials which easily recovered the factorization of n.

6 Conclusions

We have considered the small-exponent RSA problem and attacks on it using Coppersmith's method, which relies on finding short vectors in a lattice. Using theoretical and experimental observations, we have proposed a principled technique to restrict lattice basis reduction to a carefully-selected sublattice, based on the behaviour of simpler examples. This "focus group" attack specifically takes into account which parts of the lattice are likely to be used. When applied to the small-exponent RSA problem, it points to a geometric structure of the lattice in Boneh-Durfee's attack [4] that can be leveraged to reduce the running time and memory usage of the lattice basis reduction step, while allowing the attack to be applied to more RSA keys.

Several interesting questions remain for future investigations. For example, Mathematica's implementation of the L2 [17] lattice basis reduction

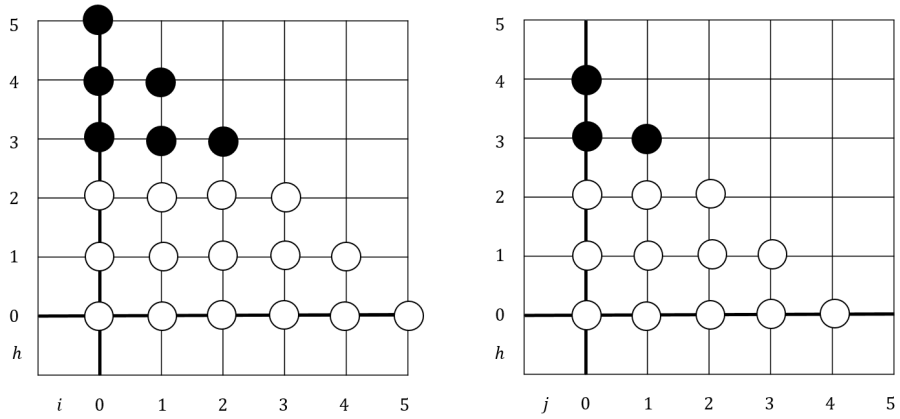


Figure 5: A representation of which polynomials from (5.2) are actually used (black circles) in solving for s and t in small examples. In this example $k = 5$ and h in (5.2) is at least $h_0 = 3$.

algorithm accounted for a several hundred bit improvement in some experiments over previous work (which instead used LLL [15]), an improvement which cannot be explained by hardware advances alone. Is it possible that special features of the lattices generated by Coppersmith's method can be exploited by new, specially designed lattice basis reduction algorithms? After all, Figure 1 suggests these lattices strongly differ from random lattices, which opens the door to such a prospect. Is it possible to specifically understand from initial principles (or perhaps even by machine learning) which parts of the lattice are not used, and perhaps redesign Coppersmith's method to include more useful vectors from the outset? Finally, might the difficult issue of algebraic independence (which is needed to establish rigorously provable results) be easier to settle using these smaller sublattices?

References

[1] Aurélie Bauer, *Vers une généralisation rigoureuse des méthodes de Coppersmith pour la recherche de petites racines de polynômes*, Ph.D. thesis, 2008.

[2] Aurélie Bauer and Antoine Joux, *Toward a Rigorous Variation of Coppersmith's Algorithm on Three Variables*, in *Advances in cryptology–EUROCRYPT 2007*, pp. 361–378, Lecture Notes in Comput. Sci., **4515**, Springer, Berlin, 2007.

[3] Daniel J. Bernstein, Yun-An Chang, Chen-Mou Cheng, Li-Ping Chou, Nadia Heninger, Tanja Lange, and Nicko Van Someren, *Factoring RSA keys from certified smart cards: Coppersmith in the wild*, in *Advances in cryptology–ASIACRYPT 2013*, pp. 341–360, Lecture Notes in Comput. Sci., **8270**, Springer, Berlin, 2013.

[4] Dan Boneh and Glenn Durfee, *Cryptanalysis of RSA with Private Key Less Than* $N^{0.292}$, IEEE Transactions on Information Theory **46**, pp. 1339–1349 (July 2000).

[5] Johannes Blömer and Alexander May, *Low Secret Exponent RSA Revisited*, in *Cryptography and Lattice Conference (CaLC 2001)*, Lecture Notes in Computer Science, **2146**, pp. 4–19, Springer-Verlag, 2001.

[6] Henry Cohn and Nadia Heninger, *Approximate common divisors via lattices*, ANTS-X (2012), The Open Book Series **1**, no. 1, pp. 271–293, (2013).

[7] Don Coppersmith, *Finding a Small Root of a Bivariate Integer Equation; Factoring with high bits known*, in *Advances in Cryptology–Eurocrypt '96*, Lecture Notes in Computer Science, **1070**, pp. 178–189. Springer-Verlag, 1996.

[8] Andrej Dujella, *A variant of Wiener's attack on RSA*, Computing **85** (2009), 77–83.

[9] The FPLLL development team, *FPLLL, a lattice reduction library.* Available at `https://github.com/fplll/fplll`

[10] Steven Galbraith, *Mathematics of Public Key Cryptography*, `http://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html`.

[11] Nick Howgrave-Graham, *Finding Small Roots of Univariate Modular Equations Revisited*, in *Proceedings of the 6th IMA International Conference on Cryptography and Coding*, pp. 131–142, Springer Verlag, 1997.

[12] Mathias Herrmann and Alexander May, *Maximizing small root bounds by linearization and applications to small secret exponent RSA,* in Proc. of PKC2010, Springer Lecture Notes in Computer Science **6056**, pp. 53–69, 2010.

[13] Ellen Jochemsz and Alexander 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, **4284**, Springer, Berlin, Heidelberg, pp. 267–282.

[14] Noburo Kunihiro, Naoyuki Shinohara, and Tetsya Izu, *A unified framework for small secret exponent attack on RSA,* in *Proc. of SAC2011*, Springer Lecture Notes in Computer Science, **7118**, pp. 260–277, 2011.

[15] Arjen K. Lenstra, Jr., Hendrik W. Lenstra, and Laszlo Lovasz, *Factoring polynomials with rational coefficients*, Mathematische Annalen, **261**, pp. 513–534, (1982).

[16] Alexander May, "New RSA Vulnerabilities Using Lattice Reduction Methods", Ph.D. thesis, 2003. `https://www.math.uni-frankfurt.de/~dmst/teaching/WS2014/Vorlesung/Alex.May.pdf`

[17] Phong Q. Nguyen and Damien Stehlé, *An LLL algorithm with quadratic complexity*, SIAM J. Comput, **39**, pp. 874–903 (2009).

[18] Ronald Rivest, Adi Shamir, and Leonard Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM, **21**, pp. 120–126 (1978).

[19] Adi Shamir, *A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem*, in *23rd annual Symposium on Foundations of Computer Science (Chicago, Ill., 1982)*, pp. 145–152, IEEE, New York, 1982.

[20] Victor Shoup, *NTL: A Library for doing Number Theory*, `http://www.shoup.net/ntl/`.

[21] M. Wiener, *Cryptanalysis of short RSA secret exponents*, IEEE Transactions on Information Theory, **36**, pp. 553–558 (1990).

[22] David Wong, `https://github.com/mimoo/RSA-and-LLL-attacks`