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

August 30, 2017

Abstract

We present a principled technique for reducing the matrix size in some applications of Coppersmith's lattice method for finding roots of modular polynomial equations. It relies on an analysis of the actual performance 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, it reduces lattice dimensions and consequently running times (sometimes by factors of two or more). We also argue that existing metrics (such as enabling condition bounds) are not as important as often thought for measuring the true performance of attacks based on Coppersmith's method. Finally, experiments are given to indicate that certain lattice reductive algorithms (such as Nguyen-Stehlé's L2) may be particularly well-suited for Coppersmith's method.

1 Introduction

Ever since Shamir's devastating attack on the Knapsack cryptosystem [S], lattice reduction algorithms such as [LLL] have had surprising success against

[*]Supported by NSF grant CNS-1526333.

cryptosystems that *a priori* have nothing to do with lattices. A fundamental example is the RSA cryptosystem [RSA], 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 lattice reduction techniques of Coppersmith [C] to discover the factorization of n.

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

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

(see Section 2 for more background on RSA). Wiener [Wi] 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 reduction, namely for 2-dimensional lattices. Boneh-Durfee [BD] apply Coppersmith's technique 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 must produce enough information to factor n (subject to an algebraic independence condition). It is an important open problem to improve the bound (1.2), which still stands as the current record despite many attempts to improve it.

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 $1-2^{-1/2} \approx .292$ [BD, BM, Wo]. The reason for this is that (1.2) is an asymptotic estimate that requires very large lattices. The difficulty of finding short vectors in lattices of dimension > 500 already serves as the hard underlying problem behind other cryptosystems. Unfortunately, the Boneh-Durfee bound does not exceed $\delta = .278$ for lattices of dimension ≤ 500. Indeed, $\delta = .278$ is close to the limit of known experiments.

Thus implementing lattice-based attacks can itself face impractically difficult problems. It is therefore natural to ask the following questions.

Questions: Can one practically solve small-exponent RSA instances for δ significantly larger than the experiments reported in [BD, BM, Wo]?

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?

How can Coppersmith's method be modified to reduce the size of the matrices involved?

The main contribution of this paper is to introduce a method to cut down the matrix size, which pushes back the choke point that lattice reduction algorithms face in large dimensions. All the computations here (unless otherwise noted) were performed in Mathematica[1] v.11 on a Surface Book laptop, and in particular did not use specialized lattice reduction packages such as [NTL]. Our main experimental finding is that it takes about an hour to factor RSA moduli $n \leq 2^{10,000}$ when $\delta \leq .277$. Though the scope of this paper is limited to small-exponent RSA, our methods appear to be applicable to other lattice problems (in particular, applications of Coppersmith's method).

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.

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 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

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

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

[1] Mathematica's `LatticeReduce` command uses a variant of the L2 algorithm of Nguyen and Stehlé [NS].

sides by $k\phi(n)$ 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 [Wi] observed that if $\delta < \frac{1}{4}$, the fraction $\frac{k}{d}$ approximates $\frac{e}{n}$ by much better 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 efficiently computed.

Following [BD], 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

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

Coppersmith's method is used to promote the polynomial congruence relation (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, let

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

for

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

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, and so 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 small enough, it must actually vanish.

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

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

where w_h is the number of nonzero monomials in $h(\cdot,\cdot)$, then an application of Cauchy-Schwartz shows $|h(x_0, y_0)| < e^m$. In particular, (x_0, y_0) is a root of $h(\cdot,\cdot)$ over $\mathbb{Z}$ since $h(x_0, y_0) \equiv 0 \pmod{e^m}$. 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 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 [BM, HM, KSI] for other attacks obtaining exponents of this size, or close to it.

Under the enabling condition (2.10), the two shortest vectors outputted by LLL are 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, a rigorous proof of algebraic independence has only recently been announced by Bauer and Joux, who previously established algebraic independence in some related contexts [BJ]. In principle (as does happen in related examples) the shortest vectors may all result in polynomials which are trivial multiples of each other, and hence not give enough equations to reveal the two unknowns x_0 and y_0.

It is worth mentioning other lattice attacks that use different polynomials than (2.3) and which also consistently beat Wiener's $\delta < 1/4$ bound. Bauer's thesis [B, 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 remaining part of the continued fraction approximation. 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 that vanish at the desired roots. Alternatively, one can instead apply the lattice method of [JM] to this partial continued fraction approach (as we have attempted in earlier experiments) – its analogous enabling condition holds for $\delta < 1/3$. Both ranges extend much further than Boneh-Durfee's $\delta < 1-2^{-1/2} \approx .292$ range, and both can be further improved using the "focus group" methodology in Section 4 below.

Despite the promising increase in this range for δ, neither of these approaches gets above .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 is reached. That calls into question the direct relevance of the enabling condition itself, and demonstrates the importance of a better understanding of the actual performance of these attacks.

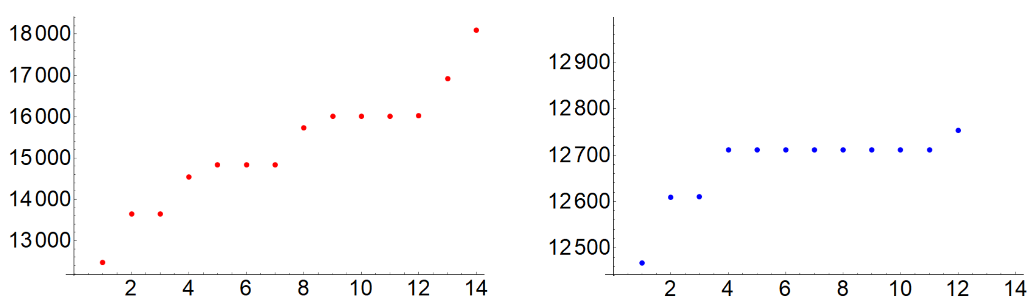

Figure 1: Logarithms of lengths of lattice basis vectors in the Boneh-Durfee .284 attack with $n \approx 2^{6,000}$ and $\delta \approx .284$, before and after lattice reduction.

We conclude this section with a remark 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.) 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 the overall lengths do differ significantly in each plot. The plot the left indicates that the input basis has several different regimes, owing to the structure of (2.6). The plot on the right shows that the output basis also has vectors in 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.

3 Some potential misconceptions about why Coppersmith's method works

Key to all theoretical analyses of Coppersmith's method is the "enabling condition" bound on the lattice determinant (e.g., (2.10)), since it provably guarantees short vectors. However, we have mentioned above that the theoretical guarantees typically do not kick in (at least in the example of small-exponent RSA) until the key sizes are much larger than found in practice. Since such theoretical analysis may in practice be describing irrelevant regimes, it is worthwhile to investigate what actually makes Coppersmith's method successful. For example, is there inhomogeneity in the geometry of the lattice that effectively reduces its dimension?

In this section, we examine a few superficial aspects of Coppersmith's method which, despite perhaps wide belief to the contrary, do not necessarily help the method work.

3.1 Are shortest vectors really key to Coppersmith?

The condition (2.9) points to the importance of short vectors in the lattice Λ, since those polynomials are small on the congruence solution (x_0, y_0) (and hence actual solutions over $\mathbb{Z}$ if the value is small enough). Not surprisingly, implementations of Coppersmith's method nearly always discard all but the shortest vectors in the output of a lattice reduction algorithm (such as LLL). However, will shortly see that this sometimes throws the solution (x_0, y_0) away with it.

Common practice notwithstanding, there are several arguments that can be made for deemphasizing the role of the shortest vector:

- **Algebraic independence.** In the multivariate setting one further requires additional, algebraically independent polynomials in order to extract the common root. In many settings (such as in the example below) the second-shortest polynomial is an obvious multiple of the shortest one. For example, there may be many small-monomial multiples of the shortest polynomial among the first few vectors of the lattice reduction output. In order to get algebraic independence, some type of mixing must occur to get a genuinely new relation, and such entanglement is often impossible without liberalizing the norm criterion.

  From the theoretical point of view, it is not hard to come up with an "attack" that beats the Boneh-Durfee .292 bound if one blindly assumes algebraic independence. (Indeed, as we discussed near the end of Section 2, Bauer's thesis [B] shows the range $\delta < .34$ may be asymptotically achieved if one only cares about producing polynomials which vanish at the desired root.) Hence reporting a range in which the enabling condition holds is only an upper bound on the range of validity of an attack: it only measures at what point the ability to find short vectors ceases. Algebraic independence may break down before this point. See [BJ] for some techniques which guarantee algebraic independence.

- **Many vectors have similar length.** On a typical lattice, LLL typically outputs several vectors of roughly comparable length, at least at the logarithmic scale. Thus the distinction of *shortest* may not be particularly significant. It also follows that one should attempt to understand the geometry of the input and output lattices to see if the outputs have drastically different length scales (see Figure 1 and the concluding remarks of Section 2).

- **Length in itself is not the right 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 there may be some loss. One might imagine leveraging some known geometry from aspects of x_0 and y_0 (such as their sign) that is known in advance.

- **An example where long vectors help.** Here is an example where longer vectors afford more algebraic independence than shorter ones.[2] Consider the 1000 bit RSA modulus $n = pq$ and key given by

$$\begin{aligned} p = {} & 32753424837507631708364161137653405626435881126097611145 4\\ & 74346957987465365057726621136658502689027080215910507483 2\\ & 0984215116927258714434174724054953133\,, \end{aligned}$$

$$\begin{aligned} q = {} & 327462704072360233831723075103626846066746692190298143145\\ & 154087005180715732984190358817594057449905589163120424047\\ & 4172883400239374471379393571624577657\,, \end{aligned}$$

and

$$\begin{aligned} d = {} & 300147077152565471186517713474704374146330287118250537992\\ & 7435326735028048350149451 = n^{\delta}, \quad \delta \approx .2707. \end{aligned}$$

We applied the BKZ reduction from [NTL] with block size 3 to the lattice from Boneh-Durfee's ".292" attack with the $(m,t) = (5,2)$. Of the 25 output vectors, only one of them (the fifth longest!) 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).

Several lessons can be drawn from this example. For instance, it is a mistake to discard the longer vectors, since they often have much more useful information than the shortest vectors. Also, the quality of the reduced basis depends greatly on the actual reduction algorithm used. At the philosophical level, this shows Coppersmith's level is not purely about vector length – it can work even when short vectors are not helpful.

3.2 How important is minimizing $|\Lambda|$?

In order to leverage provable guarantees that a lattice reduction algorithm will find a sufficiently short vector, lattices in variants of Coppersmith's attack are often modified in order to keep the covolume $|\Lambda|$ small. While this allows for rigorous analysis, there are geometric reasons why it may not be algorithmically helpful:

- 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.

[2] A similar feature was observed in A. Bauer's Ph.D. thesis [B].

- For example, suppose one appends a very long vector to a basis perpendicular to it. This would magnify the covolume without affecting the performance of LLL at all. Thus covolume by itself is a red herring.
- The ultimate goal in Coppersmith's method is not to reduce the covolume, but to increase the likelihood of finding a short vector. It's more important to identify sublattices having short vectors, which is not well-measured by the covolume.
- As we have noted in our discussion of Bauer's thesis [B] near the end of Section 2, attacks with very different enabling condition bounds may perform similarly in practice, since algebraic dependence kicks in before the enabling condition is reached. Thus $|\Lambda|$ itself may not actually enter into a meaningful bound anyhow.

An approach to identifying sublattices having small vectors is given in the next section.

4 "Focus group" attacks

We discussed above how applying lattice reduction to a sublattice may increase the chances of finding short vectors, while of course simultaneously decreasing its run time. In this section we describe a principled, evidence-based approach to selecting a sublattice in certain lattice 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 5 to small-exponent RSA.

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

1. **Set small parameters.** Find a regime which keeps the lattice dimension constant, but reduces the size of the coordinates of the basis vectors. This ensures that the lattice entries are small, which makes it faster (or even possible) to execute lattice 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).

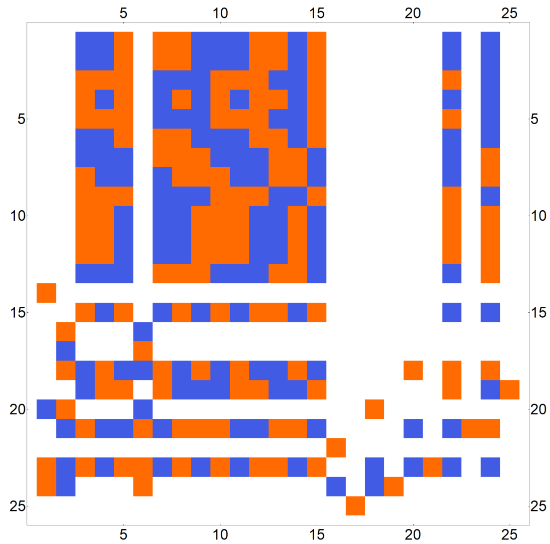

Figure 2: A representation of the change of basis matrix for the lattice reduction step in Boneh-Durfee's .284 attack (see the text for more details). The matrix has a number of columns with many zero entries (marked white).

2. **Check the output to see which parts of the original basis were actually used.** Figure 2 shows the change of basis matrix for the lattice reduction in Boneh-Durfee's .284 attack for a 6,000-bit RSA modulus n, $\delta = .251$, and $(m, t) = (4, 2)$ (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 orange (negative), blue (positive), 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. In fact, the figure indicates that most of the y-shifts are not used. It is more striking that some of the smaller x-

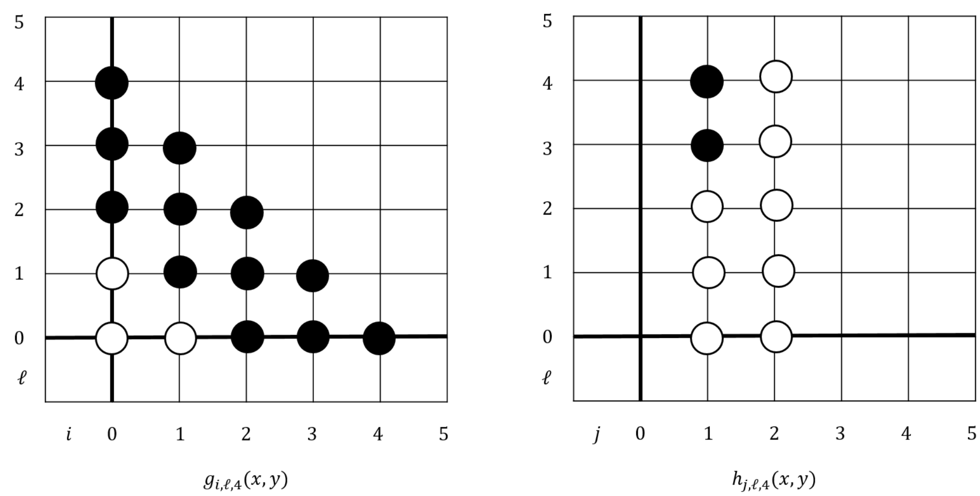

Figure 3: A representation of which polynomials in (2.6)-(2.7) are actually used (black circles) in forming the shortest vector in the lattice reduction step for a particular instance of the Boneh-Durfee attack. The unfilled, white circles represent discarded basis vectors (see the text for more details).

shifts are not used, confirming the utility of a similar device in [BM, §4]. Similar patterns arise for larger parameter sizes and were used to formulate the attack in Section 5. Indeed, examples of patterns yield useful descriptions in terms of parameters, which are then used to extrapolate good guesses for what 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 reduction on smaller lattices improves dramatically.

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

We now specialize the methodology of Section 4 to small-exponent RSA. Trials of the Boneh-Durfee .284 attack [BD] with small parameters suggest only a particular sublattice will be used, 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 determinant. Later work by Blömer and May [BM] suggests removing some of the $g_{i,\ell,m}$ as well (see also [HM, KSI]).

Our approach is guided by which vectors are likely to contribute to a nontrivial solution. 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 $\ell - 2j \leq \tau$. That is, the polynomials in (2.6) are taken for indices

$$0 \leq \ell \leq m,\ \max(-1, \sigma - \ell) < i \leq m - \ell, \text{ and } 1 \leq j \leq \min(t, 1 + \tfrac{\ell - \tau}{2}) \quad (5.1)$$

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)).

We used Mathematica v.11 on a Microsoft Surface Book with an i7-6600 CPU equipped with 16GB RAM. We did not seriously attempt to optimize the lattice reduction computations, relying instead on Mathematica's `LatticeReduce` command (which is an implementation of [NS]). In order to keep a comparison with the experiments in [BD, BM, Wo] we restricted our attention to computations that took roughly an hour or less. Our results are presented in Table 1, and include a comparison with an implementation of Boneh-Durfee's .292 attack using Mathematica on the same machine.[3] It would be interesting to perform a similar comparison with the algorithm of [BM], whose sublattice is more similar to the one selected by the "focus group" attack. The attack in [BM] satisfies the enabling condition for the same $\delta < 1 - \sqrt{1/2} \approx .292$ range as Boneh-Durfee's attack [BD]. We have not rigorously analyzed at what point our enabling condition breaks down, as it is likely moot: algebraic independence may be lost before that point anyhow (see the comments at the end of Section 2).

The size of d in the last entry in Table 1 is 220 bits longer than achieved in [BD] for a 10,000-bit RSA modulus n. However, this is mostly explained by algorithmic improvements in lattice reduction: Mathematica's `LatticeReduce` command uses the L2 algorithm [NS], which typically outperformed the BKZ implementation from [NTL] in our tests. Indeed, it is for this reason that our exponents are significantly higher than those reported in earlier experiments

[3]Note that the Boneh-Durfee .292 attack already selects a sublattice, so the performance gain against the .284 attack is even greater. The dimensions of the comparable Boneh-Durfee .292 attack lattices are slightly higher than ours: this is because their attack failed for smaller lattice sizes.

bits of n	bits of d	δ	m	t	σ	τ	Matrix size	Running Time	Comparable BD times
4000	1092	.273	6	2	2	0	28×42	1 minute	2 minutes (34 dimensional)
6000	1662	.277	8	3	2	-1	54×72	32 minutes	123 minutes (57 dimensional)
10000	2600	.260	3	1	1	0	8×14	1.58 seconds	3.21 seconds (11 dimensional)
10000	2650	.265	4	1	1	0	14×20	11 seconds	19 seconds (17 dimensional)
10000	2770	.277	8	3	2	-1	54×72	61 minutes	101 minutes (57 dimensional)

Table 1: Results of trials of the "focus group" attack on small-exponent RSA. The last column also lists the run time of an implementation of the Boneh-Durfee .292 attack for the same δ and similar lattice size (but on the same machine using the same lattice reduction algorithm). For comparison to earlier works, times refer to the lattice reduction step only.

(e.g., [BD, BM, Wo]). It is interesting to speculate whether certain features of [NS] are particularly helpful to the lattices produced by Coppersmith's method, and if so, how to leverage them further (see also the example and comments at the end of Section 3.1).

6 Conclusions

We have considered the small-exponent RSA problem and attacks on it using Coppersmith's method, which relies finding short vectors in a lattice. Using theoretical and experiment observations, we proposed a principled technique to restrict lattice reduction to carefully-selection portions of the lattice, based on the behaviour of simpler examples. This is illustrated by our "focus group" attack, which 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 [BD] that can be leveraged to halve the running time of the lattice reduction step.

Several interesting questions remain for future investigations. For example, the use of the L2 [NS] lattice reduction algorithm instead of [LLL] accounted for a several hundred bit improvement in some experiments. Is is possible that special features of the lattices generated by Coppersmith's method can be exploited by new, specially designed lattice reduction algorithms? After all, Figure 1 suggests these lattices strongly differ from random lattices, which opens the door to such a possibility. Is it possible to specifically understand from initial principles which parts of the lattice are not used, and perhaps redesign Coppersmith's method to include more useful vectors from the outset?

References

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

[BJ] Aurélie Bauer and Antoine Joux, *A Rigorous Variation of Coppersmith's Algorithm: Boneh-Durfee's Attack Revisited* ,in *Advances in cryptology–EUROCRYPT 2007*, pp. 361–378, Lecture Notes in Comput. Sci., **4515**, Springer, Berlin, 2007.

[BD] 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).

[BM] 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.

[C] 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.

[H-G] 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.

[HM] 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.

[JM] 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.

[KSI] 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.

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

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

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

[RSA] 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.

[S] 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.

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

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