Implicit Factoring: On Polynomial Time Factoring Given Only an Implicit Hint [*]

Alexander May, Maike Ritzenhofen

Horst Görtz Institute for IT-security
Faculty of Mathematics
Ruhr-University of Bochum, 44780 Bochum, Germany

alex.may@ruhr-uni-bochum.de
maike.ritzenhofen@ruhr-uni-bochum.de

Abstract. We address the problem of polynomial time factoring RSA moduli $N_1 = p_1q_1$ with the help of an oracle. As opposed to other approaches that require an oracle that *explicitly* outputs bits of p_1, we use an oracle that gives only *implicit* information about p_1. Namely, our oracle outputs a different $N_2 = p_2q_2$ such that p_1 and p_2 share the t least significant bits. Surprisingly, this implicit information is already sufficient to efficiently factor N_1, N_2 provided that t is large enough. We then generalize this approach to more than one oracle query.

Key words: Factoring with an oracle, lattices

1 Introduction

Factoring large integers is one of the most fundamental problems in algorithmic number theory and lies at the heart of RSA's security. Consequently, since the invention of RSA in 1977 [18] there have been enormous efforts for finding efficient factorization algorithms. The Quadratic Sieve [16], the Elliptic Curve Method [9] and eventually the Number Field Sieve [10] have led to a steady progress in improving the factorization complexity. However, since 1993 there is little progress from the complexity theoretic point of view when using classical Turing machines as the model of computation.

Shor's algorithm from 1994 [19] demonstrates that the factorization problem is polynomial time solvable on quantum Turing machines. Nowadays, it seems to be highly unclear whether these machines can ever be realized in practice.

The so-called *oracle complexity* of the factorization problem was first studied at Eurocrypt 1985 by Rivest and Shamir [17], who showed that $N = pq$ can be factored given an oracle that provides an attacker with bits of one of the prime

[*] The research leading to these results was supported by the German Research Foundation (DFG) as part of the project MA 2536/3-1 and has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number ICT-2007-216646 - European Network of Excellence in Cryptology II (ECRYPT II)

factors. The task is to factor in polynomial time by asking as few as possible queries to the oracle. Rivest and Shamir showed that $\frac{3}{5}\log p$ queries suffice in order to factor efficiently.

At Eurocrypt 1992, Maurer [12] allowed for an oracle that is able to answer any type of questions by YES/NO answers. Using this powerful oracle, he showed that $\epsilon \log p$ oracle queries are sufficient for any $\epsilon > 0$ in order to factor efficiently. At Eurocrypt 1996, Coppersmith [2] in turn improved the Rivest-Shamir oracle complexity for most significant bits to $\frac{1}{2}\log p$ queries. Coppersmith used this result to break the Vanstone-Zuccherato ID-based cryptosytem [21] that leaks half of the most significant bits.

In this work, we highly restrict the power of the oracle. Namely, we allow for an oracle that on input an RSA modulus $N_1 = p_1 q_1$ outputs another different RSA modulus $N_2 = p_2 q_2$ such that p_1, p_2 share their t least significant bits. Moreover, we assume for notational simplicity that the bit-sizes of p_2, q_2 are equal to the bit-sizes of p_1, q_1, respectively.

Thus, as opposed to an oracle that *explicitly* outputs bits of the prime factor p_1, we only have an oracle that *implicitly* gives information about the bits of p_1. Intuitively, since N_2 is a hard to factor RSA modulus, it should not be possible to extract this *implicit* information. We show that this intuition is false. Namely, we show that the link of the factorization problems N_1 and N_2 gives rise to an efficient factorization algorithm provided that t is large enough.

More precisely, let q_1 and q_2 be α-bit numbers. Then our lattice-based algorithm provably factors N_1, N_2 with $N_1 \neq N_2$ in quadratic time whenever $t > 2(\alpha + 1)$. In order to give a numerical example: Let N_1, N_2 have 750-bit p_1, p_2 and 250-bit q_1, q_2. Then the factorization of N_1, N_2 can be efficiently found provided that p_1, p_2 share more than 502 least significant bits. The bound $t > 2(\alpha+1)$ implies that our first result works only for imbalanced RSA moduli. Namely, the prime factors p_i have to have bit-sizes larger than twice the bit-sizes of the q_i.

Using more than one oracle query, we can further improve upon the bound on t. In the case of $k-1$ queries, we obtain $N_2, \ldots, N_k$ different RSA moduli such that all p_i share the least t significant bits. This gives rise to a lattice attack with a k-dimensional lattice L having a short vector $\mathbf{q} = (q_1, \ldots, q_k)$ that immediately yields the factorization of all $N_1, \ldots, N_k$. For constant k, our algorithm runs in time polynomial in the bit-size of the RSA moduli. As opposed to our first result, in the general case we are not able to prove that our target vector $\mathbf{q}$ is a shortest vector in the lattice L. Thus, we leave this as a heuristic assumption. This heuristic is supported by a counting argument and by experimental results that demonstrate that we are almost always able to efficiently find the factorization.

Moreover, when asking $k-1$ queries for RSA moduli with α-bit q_i that share t least significant bits of the p_i, we improve our bound to $t \geq \frac{k}{k-1}\alpha$. Hence for a larger number k of queries our bound converges to $t \geq \alpha$, which means that the p_i should at least coincide on α bits, where α is the bit-size of the q_i. In the case of RSA primes of the same bit-size, this result tells us that $N_1 = p_1 q_1, \ldots, N_k = p_1 q_k$ with the same p_1 can efficiently be factored, which is trivially true by greatest common divisor computations. On the other hand, our result is highly non-trivial whenever the bit-sizes are not balanced.

If we do not restrict ourselves to polynomial running time, then we can easily adapt our method to also factor balanced RSA moduli. All that we have to do is to determine a small quantity of the bits of q_i by brute force search. Using these bits we can apply the previous method in order to determine at least half of the bits of all q_i. The complete factorization of *all* RSA moduli N_i is then retrieved by the aforementioned lattice-based algorithm of Coppersmith [3].

Currently, we are not aware of an RSA key generation that uses primes sharing least significant bits. The Steinfeld-Zheng system [20] uses moduli $N = pq$ such that p, q itself share least significant bits, for which our algorithm does not apply. Naturally, one application of our result is malicious key generation of RSA moduli, i.e. the construction of backdoored RSA moduli [5, 22].

Another application is a novel kind of attack on a public key generator. Suppose an attacker succeeds to manipulate those t registers of an RSA public key generator that hold the least significant bits of one prime factor such that these registers are stuck to some unknown value. E.g., take an attacker that simply destroys the registers with the help of a laser beam such that he has no control on the register's values. If the RSA key parameters are within our bounds, the attacker can easily collect sufficiently many RSA moduli that allow him to factor all of them. Thus, he uses the RSA key generator as an oracle. Notice that the RSA generator will usually not even notice such an attack since the RSA moduli look innocent.

Moreover, we feel that our algorithm will be useful for *constructive* cryptographic applications as well. Consider the task that our oracle has to solve, which we call the *one more RSA modulus problem*, i.e one has to produce on input an RSA modulus $N = pq$ other moduli $N_i = p_i q_i$ whose factors p_i share their least significant bits.

Our construction shows that this problem is for many parameter settings equivalent to the factorization problem. So the *one more RSA modulus problem* might serve as a basis for various cryptographic primitives, whose security is then in turn directly based on factoring (imbalanced) integers.

In addition to potential applications, we feel that our result is of strong theoretical interest, since we show for the first time that quite surprisingly implicit information is sufficient in order to factor efficiently. In turn, this implies that already a really weak form of an oracle suffices for achieving a polynomial time factorization process. In the oracle-based line of research, reducing the number of queries and diminishing the power of the oracles is the path that leads to a better understanding of the complexity of the underlying factorization problem.

We organize our paper as follows. In Section 2, we give the necessary facts about lattices. In Section 3, we introduce our rigorous construction with one oracle query, i.e. with two RSA moduli. In Section 4, we generalize our construction to an arbitrary fixed number of queries. This makes our construction heuristic. In Section 5, we adapt our heuristic construction to the case of balanced RSA moduli. In Section 6, we experimentally confirm the validity of our heuristics.

2 Preliminaries

An integer lattice L is a discrete additive subgroup of $\mathbb{Z}^n$. An alternative equivalent definition of an integer lattice can be given via a basis.

Let $d, n \in \mathbb{N}$, $d \leq n$. Let $\mathbf{b_1}, \ldots, \mathbf{b_d} \in \mathbb{Z}^n$ be linearly independent vectors. Then the set of all integer linear combinations of the $\mathbf{b_i}$ spans an integer *lattice* L, i.e.

$$L = \left\{ \sum_{i=1}^{d} a_i \mathbf{b_i} \mid a_i \in \mathbb{Z} \right\}.$$

We call $B = \begin{pmatrix} \mathbf{b_1} \\ \vdots \\ \mathbf{b_d} \end{pmatrix}$ a *basis* of the lattice, the value d denotes the *dimension* or *rank* of the basis. The lattice is said to have *full rank* if $d = n$. The *determinant* $\det(L)$ of a lattice is the volume of the parallelepiped spanned by the basis vectors. The determinant $\det(L)$ is invariant under unimodular basis transformations of B. In case of a full rank lattice $\det(L)$ is equal to the absolute value of the Gramian determinant of the basis B.

Let us denote by $||\mathbf{v}||$ the Euclidean ℓ_2-norm of a vector $\mathbf{v}$. Hadamard's inequality [13] relates the length of the basis vectors to the determinant.

Lemma 1 (Hadamard). *Let* $B = \begin{pmatrix} \mathbf{b_1} \\ \vdots \\ \mathbf{b_n} \end{pmatrix} \in \mathbb{Z}^{n \times n}$, $n \in \mathbb{N}$, *be an arbitrary non-singular matrix. Then*

$$\det(B) \leq \prod_{i=1}^{n} ||\mathbf{b_i}|| .$$

The successive minima $\lambda_i(L)$ of the lattice L are defined as the minimal radius of a ball containing i linearly independent lattice vectors of L. In a two-dimensional lattice L, basis vectors $\mathbf{v_1}, \mathbf{v_2}$ with lengths $||v_1|| = \lambda_1(L)$ and $||v_2|| = \lambda_2(L)$ are efficiently computable via Gaussian reduction.

Theorem 1. *Let* $\mathbf{b_1}, \mathbf{b_2} \in \mathbb{Z}^n$ *be basis vectors of a two-dimensional lattice* L. *Then the Gauss-reduced lattice basis vectors* $\mathbf{v_1}, \mathbf{v_2}$ *can be determined in time* $O(\log^2(\max\{||\mathbf{v_1}||, ||\mathbf{v_2}||\})$. *Furthermore,*

$$||\mathbf{v_1}|| = \lambda_1(L) \text{ and } ||\mathbf{v_2}|| = \lambda_2(L).$$

Information on Gaussian reduction and its running time can be found in [13].

A shortest vector of a lattice satisfies the Minkowski bound, which relates the length of a shortest vector to the determinant and dimension of the lattice.

Theorem 2 (Minkowski [14]). *Let* $L \subseteq \mathbb{Z}^{n \times n}$ *be an integer lattice. Then* L *contains a non-zero vector* $\mathbf{v}$ *with*

$$||\mathbf{v}|| = \lambda_1(L) \leq \sqrt{n} \det(L)^{\frac{1}{n}}.$$

Vectors with short norm can be computed by the LLL algorithm of Lenstra, Lenstra, and Lovász [11].

Theorem 3 (LLL). *Let L be a d-dimensional lattice with basis $\mathbf{b_1}, \ldots, \mathbf{b_d} \in \mathbb{Z}^n$. Then the LLL algorithm outputs a reduced basis $\mathbf{v_1}, \ldots, \mathbf{v_d}$ with the following property:*

$$||\mathbf{v_1}|| \leq 2^{\frac{d-1}{4}} \det(L)^{\frac{1}{d}}.$$

The running time of this algorithm is $O(d^4 n(d+\log b_{\max}) \log b_{\max})$, where $b_{\max} \in \mathbb{N}$ denotes the largest entry in the basis matrix.

For a proof of the upper bound of a shortest LLL vector compare [11]. The running time is the running time of the so-called L^2-algorithm, an efficient LLL version due to Nguyen and Stehlé [15].

The LLL algorithm can be used for factoring integers with partly known factors as Coppersmith showed in [3].

Theorem 4 ([3] Theorem 5). *Let N be an n-bit composite number. Then we can find the factorization of $N = pq$ in polynomial time if we know the low order $\frac{n}{4}$ bits of p.*

3 Implicit Factoring of Two RSA Moduli

Assume that we are given two different RSA moduli $N_1 = p_1 q_1, N_2 = p_2 q_2$, where p_1, p_2 coincide on the t least significant bits. I.e., $p_1 = p + 2^t \tilde{p}_1$ and $p_2 = p + 2^t \tilde{p}_2$ for some common p that is *unknown* to us. Can we use the information that the prime factors of N_1 and N_2 share their t least significant bits without knowing these bits *explicitly*? I.e., can we factor N_1, N_2 given only *implicit* information about one of the factors?

In this section, we will answer this question in the affirmative. Namely, we will show that there is an algorithm that recovers the factorization of N_1 and N_2 in quadratic time provided that t is sufficiently large.
We start with

$$(p + 2^t \tilde{p}_1) q_1 = N_1$$
$$(p + 2^t \tilde{p}_2) q_2 = N_2.$$

These two equations contain five unknowns p, p_1, p_2, q_1 and q_2. By reducing both equations modulo 2^t, we can eliminate the two unknowns $\tilde{p}_1, \tilde{p}_2$ and get

$$pq_1 \equiv N_1 \bmod 2^t$$
$$pq_2 \equiv N_2 \bmod 2^t.$$

Since q_1, q_2 are odd, we can solve both equations for p. This leaves us with $\frac{N_1}{q_1} \equiv \frac{N_2}{q_2} \bmod 2^t$, which we write in form of the *linear* equation

$$(N_1^{-1} N_2) q_1 - q_2 \equiv 0 \bmod 2^t. \tag{1}$$

The set of solutions

$$L = \{(x_1, x_2) \in \mathbb{Z}^2 \mid (N_1^{-1}N_2)x_1 - x_2 \equiv 0 \bmod 2^t\}$$

forms an additive, discrete subgroup of $\mathbb{Z}^2$. Thus, L is a 2-dimensional integer lattice. L is spanned by the row vectors of the basis matrix

$$B_L = \begin{pmatrix} 1 & N_1^{-1}N_2 \\ 0 & 2^t \end{pmatrix}.$$

Let us briefly check that the integer span of B_L, denoted by $\text{span}(B_L)$, is indeed equal to L. First, $\mathbf{b_1} = (1, N_1^{-1}N_2)$ and $\mathbf{b_2} = (0, 2^t)$ are solutions of $(N_1^{-1}N_2)x_1 - x_2 = 0 \bmod 2^t$. Thus, every integer linear combination of $\mathbf{b_1}$ and $\mathbf{b_2}$ is a solution which implies that $\text{span}(B_L) \subseteq L$.

Conversely, let $(x_1, x_2) \in L$, i.e. $(N_1^{-1}N_2)x_1 - x_2 = k \cdot 2^t$ for some $k \in \mathbb{Z}$. Then $(x_1, -k)B_L = (x_1, x_2) \in \text{span}(B_L)$ and thus $L \subseteq \text{span}(B_L)$.

Notice that by Eq. (1), we have $(q_1, q_2) \in L$. If we were able to find this vector in L then we could factor N_1, N_2 easily. Let us first provide some intuition under which condition the vector $\mathbf{q} = (q_1, q_2)$ is a short vector in L. We know that the length of the shortest vector is upper bounded by the Minkowski bound $\sqrt{2}\det(L)^{\frac{1}{2}} = \sqrt{2} \cdot 2^{\frac{t}{2}}$.

Since we assumed that q_1, q_2 are α-bit primes, we have $q_1, q_2 \leq 2^\alpha$. If α is sufficiently small, then $||\mathbf{q}||$ is smaller than the Minkowski bound and, therefore, we can expect that $\mathbf{q}$ is among the shortest vectors in L. This happens if

$$||\mathbf{q}|| \leq \sqrt{2} \cdot 2^\alpha \leq \sqrt{2} \cdot 2^{\frac{t}{2}}.$$

So if $t \geq 2\alpha$ we expect that $\mathbf{q}$ is a short vector in L. We can find a shortest vector in L using Gaussian reduction on the lattice basis B in time $\mathcal{O}(\log^2(2^t)) = \mathcal{O}(\log^2(\min\{N_1, N_2\}))$. Hence, under the heuristic assumption that $\mathbf{q} = (q_1, q_2)$ is a shortest vector in L we can factor N_1, N_2 in quadratic time. Under a slightly more restrictive condition, we can completely remove the heuristic assumption.

Theorem 5. *Let $N_1 = p_1q_1, N_2 = p_2q_2$ be two different RSA moduli with α-bit q_i. Suppose that p_1, p_2 share at least $t > 2(\alpha + 1)$ bits. Then N_1, N_2 can be factored in quadratic time.*

Let

$$B_L = \begin{pmatrix} 1 & N_1^{-1}N_2 \\ 0 & 2^t \end{pmatrix}$$

be the lattice basis defined as before.

B_L spans a lattice L with shortest vector $\mathbf{v}$ that satisfies

$$||\mathbf{v}|| \leq \sqrt{2}\det(L)^{\frac{1}{2}} = 2^{\frac{t+1}{2}}.$$

Performing Gaussian reduction on B_L, we get an equivalent basis $B = \begin{pmatrix} \mathbf{b_1} \\ \mathbf{b_2} \end{pmatrix}$ such that

$$||\mathbf{b_1}|| = \lambda_1(L) \text{ and } ||\mathbf{b_2}|| = \lambda_2(L).$$

Our goal is to show that $\mathbf{b_1} = \pm\mathbf{q} = \pm(q_1, q_2)$ which is sufficient for factoring N_1 and N_2.

As L is of full rank, by Hadamard's inequality we have

$$||\mathbf{b_1}|| \, ||\mathbf{b_2}|| \geq \det(L).$$

This implies

$$||\mathbf{b_2}|| \geq \frac{\det(L)}{||\mathbf{b_1}||} = \frac{\det(L)}{\lambda_1(L)}.$$

Substituting $\det(L) = 2^t$ and using $\lambda_1(L) \leq 2^{\frac{t+1}{2}}$ leads to

$$||\mathbf{b_2}|| \geq \frac{2^t}{2^{\frac{t+1}{2}}} = 2^{\frac{t-1}{2}}.$$

This implies for any lattice vector $\mathbf{v} = a_1\mathbf{b_1} + a_2\mathbf{b_2}$ with $||\mathbf{v}|| < 2^{\frac{t-1}{2}}$ that $a_2 = 0$, as otherwise $\lambda_2(L) \leq ||\mathbf{v}|| < ||\mathbf{b_2}||$ which contradicts the optimality of $\mathbf{b_2}$ from Theorem 1. Thus, every $\mathbf{v}$ with $||\mathbf{v}|| < 2^{\frac{t-1}{2}}$ is a multiple of $\mathbf{b_1}$. Notice that $\mathbf{q} = (q_1, q_2) \in L$ fulfills $||\mathbf{q}|| = \sqrt{2} \cdot 2^\alpha = 2^{\frac{2\alpha+1}{2}}$. Consequently, we have $||\mathbf{q}|| < ||\mathbf{b_2}||$ if

$$2^{\frac{2\alpha+1}{2}} < 2^{\frac{t-1}{2}} \Leftrightarrow 2(\alpha+1) < t$$

Therefore, we get $\mathbf{q} = a\mathbf{b_1}$ for some $a \in \mathbb{Z} - \{0\}$. Let $\mathbf{b_1} = (b_{11}, b_{12})$, then $\gcd(q_1, q_2) = \gcd(ab_{11}, ab_{12}) \geq a$. But q_1, q_2 are primes and wlog $q_1 \neq q_2$, since otherwise we can factor N_1, N_2 by computing $\gcd(N_1, N_2)$. Therefore, $|a| = 1$ and we obtain $\mathbf{q} = \pm\mathbf{b_1}$, which completes the factorization.

The running time of the factorization is determined by the running time of the Gaussian reduction, which can be performed in $\mathcal{O}(t^2) = \mathcal{O}(\log^2(\min\{N_1, N_2\}))$ steps. □

4 Implicit Factoring of k RSA Moduli

The approach from the previous section can be generalized to an arbitrary fixed number $k-1$ of oracle queries. This gives us k different RSA moduli

$$\begin{aligned} N_1 &= (p + 2^t\tilde{p}_1)q_1 \\ &\vdots \\ N_k &= (p + 2^t\tilde{p}_k)q_k \end{aligned} \tag{2}$$

with α-bit q_i.

We transform the system of equations into a system of k equations modulo 2^t

$$\begin{aligned} pq_1 - N_1 &\equiv 0 \pmod{2^t} \\ &\vdots \\ pq_k - N_k &\equiv 0 \pmod{2^t} \end{aligned}$$

in $k+1$ variables.

Analogous to the two equation case, we solve each equation for p. This can be done because all the q_i are odd. Thus, we get $\frac{N_1}{q_1} = \frac{N_i}{q_i} \pmod{2^t}$ for $i = 2, \ldots, k$. Writing this as $k-1$ linear equations gives us:

$$\begin{aligned} N_1^{-1}N_2q_1 - q_2 &\equiv 0 \pmod{2^t} \\ &\vdots \\ N_1^{-1}N_kq_1 - q_k &\equiv 0 \pmod{2^t}. \end{aligned}$$

With the same arguments as in the preceding section the set

$$L = \{(x_1, \ldots, x_k) \in \mathbb{Z}^k \mid N_1^{-1}N_ix_1 - x_i \equiv 0 \pmod{2^t} \text{ for all } i = 2, \ldots, k\}$$

forms a lattice. This lattice L is spanned by the row vectors of the following basis matrix

$$B_L = \begin{pmatrix} 1 & N_1^{-1}N_2 & \cdots & & N_1^{-1}N_k \\ 0 & 2^t & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 2^t \end{pmatrix}.$$

Note that $\mathbf{q} = (q_1, \ldots, q_k) \in L$ has norm $||\mathbf{q}|| \le \sqrt{k}2^\alpha$. We would like to have $||\mathbf{q}|| = \lambda_1(L)$ as in Section 3. The length $\lambda_1(L)$ of a shortest vector in L is bounded by

$$\lambda_1(L) \le \sqrt{k}(\det(L))^{\frac{1}{k}} = \sqrt{k}(2^{t(k-1)})^{\frac{1}{k}}.$$

Thus, if $\mathbf{q}$ is indeed a shortest vector then

$$||\mathbf{q}|| = \sqrt{k}2^\alpha \le \sqrt{k} \cdot 2^{t\frac{k-1}{k}}. \tag{3}$$

This implies the condition $t \ge \frac{k}{k-1}\alpha$. We make the following heuristic assumption.

Assumption 6 *Let $N_1, \ldots, N_k$ be as defined in Eq. (2) with $t \ge \frac{k}{k-1}\alpha$. Further, let $\mathbf{b_1}$ be a shortest vector in L. Then $\mathbf{b_1} = \pm(q_1, \ldots, q_k)$.*

Theorem 7. *Let $N_1, \ldots, N_k$ be as defined in Eq.* (2) *with $t \geq \frac{k}{k-1}\alpha$. Under Assumption 6, we can find the factorization of all $N_1, \ldots, N_k$ in time polynomial in $(k^{\frac{k}{2}}, \max_i\{\log N_i\})$.*

We show the validity of Assumption 6 experimentally in Section 6.

The running time is determined by the time to compute a shortest vector in L [8, 7]. This implies that for any lattice L of rank k such that $k^{\frac{k}{2}} = \text{poly}(\max_i\{\log N_i\})$, i.e. especially for lattices with fixed rank k, we can compute the factorization of all N_i in time polynomial in their bit-size.

For large k, our bound converges to $t \geq \alpha$. This means that the amount t of common least significant bits has to be at least as large as the bit-size of the q_i. In turn, this implies that our result only applies to RSA moduli with different bit-sizes of p_i and q_i. On the other hand, this is the best result that we could hope for in our algorithm. Notice that we construct the values of the q_i by solving equations modulo 2^t. Thus, we can fully recover the q_i only if their bit-size α is smaller than t. In the subsequent section, we will overcome this problem by avoiding the full recovery of all q_i, which in turn leads to an algorithm for balanced RSA moduli.

Remark: All of our results still hold if 2^t is replaced by an arbitrary modulus $M \geq 2^t$. We used a power of two only to illustrate our results in terms of bits.

5 Implicit Factoring of Balanced RSA Moduli

We slightly adapt the method from Section 4 in order to factor balanced n-bit integers, i. e. $N_i = p_i q_i$ such that p_i and q_i have bitsize $\frac{n}{2}$ each. The modification mainly incorporates a small brute force search on the most significant bits.

Assume that we are given k RSA moduli as in (2). From these moduli we derive $k-1$ linear equations in k variables:

$$\begin{aligned} N_1^{-1}N_2q_1 - q_2 &\equiv 0 \pmod{2^t} \\ &\vdots \\ N_1^{-1}N_kq_1 - q_k &\equiv 0 \pmod{2^t} \end{aligned}$$

The bitsize of the q_i is now fixed to $\alpha = \frac{n}{2}$ which is equal to the bitsize of the p_i, i. e. now the number t of bits on which the p_i coincide has to satisfy $t \leq \alpha$. In the trivial case of $t = \alpha = \frac{n}{2}$ we can directly factor the N_i via greatest common divisor computations as then $p_i = p$ for $i = 1, \ldots, k$.

Thus, we only consider $t < \frac{n}{2}$. With a slight modification of the method in Section 4, we compute all $q_i \pmod{2^t}$. Since $t < \frac{n}{2}$, this does not give us the q_i directly, but only their t least significant bits. But if $t \geq \frac{n}{4}$, we can use Theorem 4 for finding the full factorization of each N_i in polynomial time. In order to minimize the time complexity, we assume $t = \frac{n}{4}$ throughout this section.

To apply Theorem 7 of Section 4 the bit-size of the q_i has to be smaller than $\frac{k-1}{k}t$. Thus, we have to guess roughly $\frac{1}{k} \cdot t = \frac{n}{4k}$ bits for each q_i. Since we consider k moduli, we have to guess a total number of $\frac{n}{4}$ bits. Notice that this is the same amount of bits as for guessing one half of the bits of *one* q_j, which in turn allows to efficiently find this q_j using Theorem 4. With a total amount of $\frac{n}{4}$ bits however, our algorithm will allow us to efficiently find *all* q_i, $i = 1, \ldots, k$.

Let us describe our modification more precisely. We split $q_i \pmod{2^{\frac{n}{4}}}$ into $2^\beta \tilde{q}_i + x_i \pmod{2^{\frac{n}{4}}}$. The number β depends on the number of oracle calls $k-1$ such that the condition $\beta < \frac{(k-1)}{k} \cdot \frac{n}{4}$ holds. We therefore choose β to be the largest integer smaller than $\frac{(k-1)n}{4k}$. This implies that the $x_i \leq 2^\beta$ are small enough to be determined analogous to Section 4, provided that the $\tilde{q}_i$ are known. As discussed before, in practice we can guess an amount of $\frac{n}{4k}$ bits for determining each $\tilde{q}_i$, or we can find these bits by other means, e.g. by side-channel attacks.

Suppose now that the $\tilde{q}_i$ are given for each i. We obtain the following set of equations

$$\begin{aligned} N_1^{-1}N_2x_1 - x_2 &\equiv 2^\beta(\tilde{q_2} - N_1^{-1}N_2\tilde{q_1}) \pmod{2^{\frac{n}{4}}} \\ &\vdots \\ N_1^{-1}N_kx_1 - x_k &\equiv 2^\beta(\tilde{q_k} - N_1^{-1}N_k\tilde{q_1}) \pmod{2^{\frac{n}{4}}}. \end{aligned} \tag{4}$$

Let $c_i = 2^\beta(\tilde{q_i} - N_1^{-1}N_i\tilde{q_1})$, $i = 2, \ldots, k$, denote the known right-hand terms. In contrast to Section 4, the equations (4) that we have to solve are inhomogenous. Let us first consider the lattice L that consists of the homogenous solutions

$$L = \{(x_1, \ldots, x_k) \in \mathbb{Z}^k \mid N_1^{-1}N_ix_1 - x_i \equiv 0 \pmod{2^{\frac{n}{4}}}, i = 2, \ldots, k\}.$$

L is spanned by the rows of the following basis matrix

$$B_L = \begin{pmatrix} 1 & N_1^{-1}N_2 & \cdots & & N_1^{-1}N_k \\ 0 & 2^{\frac{n}{4}} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 2^{\frac{n}{4}} \end{pmatrix}.$$

Let $l_i \in \mathbb{Z}$ such that $N_1N_i^{-1}x_1 + l_i2^t = x_i + c_i$. Then we let

$$\mathbf{q}' := (x_1, l_2, \ldots, l_k)B_L = (x_1, x_2 + c_2, \ldots, x_k + c_k) \in L.$$

Moreover, if we define the target vector $\mathbf{c} := (0, c_2, \ldots, c_k)$, then the distance between $\mathbf{q}'$ and $\mathbf{c}$ is

$$||\mathbf{q}' - \mathbf{c}|| = ||(x_1, \ldots, x_k)|| \leq \sqrt{k}2^\beta \leq \sqrt{k} \cdot 2^{\frac{(k-1)n}{4k}}.$$

This is the same bound that we achieved in Section 4 for the length of a shortest vector in Eq. (3) when $t = \frac{n}{4}$. So instead of solving a shortest vector problem, we have to solve a closest vector problem in L with target vector $\mathbf{c}$. Closest vectors can be found in polynomial time for fixed lattice dimension k (see Blömer [1]). We make the heuristic assumption that $\mathbf{q}'$ is indeed a closest vector to $\mathbf{c}$ in L.

Assumption 8 *Let $N_1, \dots, N_k$ be as defined in Eq. (4) with $\beta < \frac{(k-1)n}{4k}$. Further, let $\mathbf{b_1}$ be a closest vector to $\mathbf{c}$ in L. Then $\mathbf{b_1} = \pm \mathbf{q_1'}$.*

Theorem 9. *Let $N_1, \dots, N_k$ be as defined in Eq. (4) with $\beta < \frac{(k-1)n}{4k}$. Under Assumption 8, we can find the factorization of all $N_1, \dots, N_k$ in time $2^{\frac{n}{4}} \cdot$ $\mathrm{poly}(k!, \max_i\{\log N_i\})$.*

The running time is determined by the time for guessing each $\tilde{q}_i$ and the time for finding a closest vector in L.

6 About our Heuristic Assumptions

In this section we have a closer look at the two heuristics from the previous sections, Assumption 6 and Assumption 8. We first give a counting argument that supports our heuristics and then demonstrate experimentally that our constructions work very well in practice.

6.1 A Counting Argument that Supports our Assumptions

Recall that in Section 4, the lattice L consists of all solutions $\mathbf{q} = (q_1, \dots, q_k)$ of the system of equations

$$\begin{aligned} N_1^{-1} N_2 q_1 &\equiv q_2 \pmod{2^t} \\ &\vdots \\ N_1^{-1} N_k q_1 &\equiv q_k \pmod{2^t} \end{aligned} \tag{5}$$

As $\gcd(N_1^{-1}N_i, 2^t) = 1$ for any i, the mapping $f_i : x \mapsto N_1^{-1}N_i x \pmod{2^t}$ is bijective. Therefore, the value of q_1 uniquely determines the values of q_i, $i = 2, \dots, k$.

In total the system of equations has as many solutions as there are values to choose q_1 from, which is 2^t. Now suppose $q_1 \leq 2^{\frac{(k-1)t}{k}}$. How many vectors $\mathbf{q}$ do we have such that $q_i \leq 2^{\frac{(k-1)t}{k}}$ for all $i = 1, \dots, k$ and thus $||\mathbf{q}|| \leq \sqrt{k} 2^{\frac{(k-1)t}{k}}$?

Assume for each $i = 2, ..., k$ that the value q_i is uniformly distributed in $\{0, \dots, 2^t - 1\}$ and that the distributions of q_i and q_j are independent if $i \neq j$. Then the probability that $q_i \leq 2^{\frac{(k-1)t}{k}}$ is

$$\Pr\left(q_i \leq 2^{\frac{(k-1)t}{k}}\right) = \frac{2^{\frac{(k-1)t}{k}}}{2^t} = 2^{-\frac{t}{k}}.$$

Furthermore, the probability that $q_i \leq 2^{\frac{(k-1)t}{k}}$ for all $i = 2, \dots, k$ is

$$\Pr\left(q_2 \leq 2^{\frac{(k-1)t}{k}}, \dots, q_k \leq 2^{\frac{(k-1)t}{k}}\right) = \left(2^{-\frac{t}{k}}\right)^{k-1} = 2^{-\frac{(k-1)t}{k}}$$

Consequently, for a given value of $q_1 \leq 2^{\frac{(k-1)t}{k}}$ the expected number of vectors $\mathbf{q}$ such that $q_i \leq 2^{\frac{(k-1)t}{k}}$ for all $i = 1, \dots, k$ is $2^{\frac{(k-1)t}{k}} \cdot 2^{-\frac{(k-1)t}{k}} = 1$. Therefore, we expect that only one lattice vector, namely $\mathbf{q}$, is short enough to satisfy the Minkowski bound. Hence, we expect that $\pm\mathbf{q}$ is a unique shortest vector in L if its length is significantly below the bound $\sqrt{k}2^{\frac{(k-1)t}{k}}$. This counting argument strongly supports our Assumption 6.

Remark: In order to analyze Assumption 8 we can argue in a completely analogous manner. The inhomogenous character of the equations does not influence the fact that the q_i are uniquely determined by q_1.

6.2 Experiments

We verified our assumptions in practice by running experiments on a Core2 Duo 1.66GHz notebook. The attacks were implemented using Magma[1] Version 2.11. Instead of taking a lattice reduction algorithm which provably returns a basis with a shortest vector as first basis vector we have used the LLL algorithm [11], more precisely its L^2 version of Nguyen and Stehlé [15] which is implemented in Magma. Although by LLL reduction the first basis vector only approximates a shortest vector in a lattice, for our lattice bases with dimensions up to 100 LLL-reduction was sufficient. In nearly all cases the first basis vector was equal to the vector $\pm\mathbf{q} = \pm(q_1, \ldots, q_k)$, when we chose suitable attack parameters.

First, we considered the cased of imbalanced RSA moduli from Theorem 7. We chose $N_i = (p + 2^t\tilde{p}_i)q_i$, $i = 1, \ldots, k$, of bit-size $n = 1000$ with varying bitsizes of q_i. For fixed bitsize α of q_i and fixed number k of moduli, we slightly played with the parameter t of common bits close to the bound $t \geq \frac{k}{k-1}\alpha$ in order to determine the minimal t for which our heuristic is valid.

bitsize α of the q_i	no. of moduli k	bound $\frac{k}{k-1}\alpha$	number of shared bits t	success rate
250	3	375	377	0%
250	3	375	378	97%
350	10	389	390	0%
350	10	389	391	100%
400	100	405	409	0%
400	100	405	410	100%
440	50	449	452	16%
440	50	449	453	97%
480	100	485	491	38%
480	100	485	492	98%

Table 1. Attack for imbalanced RSA moduli

The running time of all experiments was below 10 seconds.

[1] http://magma.maths.usyd.edu.au/magma/

In Table 1, we called an experiment successful if the first basis vector $\mathbf{b}_1$ in our LLL reduced basis was of the form $\mathbf{b_1} = \pm\mathbf{q} = \pm(q_1, \ldots, q_k)$, i.e. it satisfied Assumption 6. There were some cases, where other basis vectors were of the form $\pm\mathbf{q}$, but we did not consider these cases.

As one can see by the experimental results, Assumption 6 only works smoothly when our instances were a few extra bits beyond the bound of Theorem 7. This is not surprising since the counting argument from Section 6.1 tells us that we loose uniqueness of the shortest vector as we approach the theoretical bound. In practice, one could either slightly increase the number t of shared bits or the number k of oracle calls for making the attack work.

Analogously, we made experiments with balanced RSA moduli to verify Assumption 8. Instead of computing closest vectors directly, we used the well-known standard embedding of a d-dimensional closest vector problem into an $(d+1)$-dimensional shortest vector problem ([6], Chapter 4.1), where the shortest vector is of the form $\mathbf{b_1} = (\mathbf{q}' - \mathbf{c}, c')$, c' constant. Since $\mathbf{c}$ and c' are known, this directly yields $\mathbf{q}'$ and therefore the factorization of all RSA moduli. For solving the shortest vector problem, we again used the LLL algorithm.

As before we called an experiment successful, if $\mathbf{b_1}$ was of the desired form, i.e. if Assumption 8 held. In our experiments we used 1000 bit N_i with a common share p of $t = 250$ bits.

no. of moduli k	bound $\lceil \frac{n}{4k} \rceil$	bits known from q_i	success rate
3	84	85	74%
3	84	86	99%
10	25	26	20%
10	25	27	100%
50	5	8	46%
50	5	9	100%

Table 2. Attack for balanced 1000-bit N_i with 250 bits shared

All of our experiments ran in less than 10 seconds. Here, we assumed that we know the required bits of each q_i, i.e. the running time does not include the factor for a brute-force search.

Similar to the experimental results for the imbalanced RSA case, our heuristic Assumption 8 works well in the balanced case, provided that we spend a few extra bits to the theoretical bound in order to enforce uniqueness of the closest vector.

References

1. J. Blömer, "Closest Vectors, Successive Minima, and Dual HKZ-bases of Lattices", ICALP 2000, Lecture Notes in Computer Science, Volume 1853, Springer-Verlag, pp.

248-259, 2000

2. D. Coppersmith, “Finding a Small Root of a Bivariate Integer Equation, Factoring with High Bits Known”, Advances in Cryptology (Eurocrypt ’96), Lecture Notes in Computer Science, Volume 1070, pp. 178-189, Springer-Verlag, 1996
3. D. Coppersmith, “Small solutions to polynomial equations and low exponent vulnerabilities”, Journal of Cryptology, Volume 10(4), pp. 223-260, 1997
4. D. Coppersmith, “Finding Small Solutions to Small Degree Polynomials”, CaLC 2001, Lecture Notes in Computer Science, Volume 2146, pages 20-31, 2001
5. C. Crépeau, A. Slakmon, “Simple Backdoors for RSA Key Generation”, Topics in Cryptology(CT-RSA 2003), Lecture Notes in Computer Science, Volume 2612, pp. 403-416, Springer-Verlag, 2003
6. D. Micciancio, S. Goldwasser, “Complexity of Lattice Problems: A cryptographic perspective”, vol. 671 of Kluwer International Series in Engineering and Computer Science, Kluwer Academic Publishers, Boston, Massachusetts, 2002
7. B. Helfrich, “Algorithms to Construct Minkowski Reduced and Hermite Reduced Lattice Basis”, Theoretical Computer Science, Volume 41, pp. 125-139, 1985
8. R. Kannan, “Minkowski’s Convex Body Theorem and Integer Programming”, Mathematics of Operations Research, Volume 12, No. 3, pp. 415-440, 1987
9. H. W. Jr. Lenstra,“Factoring Integers with Elliptic Curves”, Ann. Math. 126, pp. 649-673, 1987
10. A. K. Lenstra, H. W. Jr. Lenstra, “The Development of the Number Field Sieve”, Springer-Verlag, 1993
11. A. K. Lenstra, H. W. Lenstra, and L. Lovász, “Factoring polynomials with rational coefficients”, Mathematische Annalen, Volume 261, pp. 513-534, 1982
12. U. M. Maurer, “Factoring with an Oracle”, Advances in Cryptology (Eurocrypt ’92), Lecture Notes in Computer Science, Volume 658, pp. 429-436, Springer-Verlag 1993
13. C. D. Meyer, Matrix Analysis and Applied Linear Algebra, Cambridge University Press, 2000
14. H. Minkowski, Geometrie der Zahlen, Teubner-Verlag, 1896
15. P. Q. Nguyen, D. Stehlé, “Floating Point LLL Revisited”, In Advances in Cryptology (Eurocrypt 2005), Lecture Notes in Computer Science, Volume 3494, pages 215 - 233, Springer-Verlag, 2005
16. C. Pomerance, "The Quadratic Sieve Factoring Algorithm." In Advances in Cryptology (Eurocrypt ’84), Lecture Notes in Computer Science, Volume 209, pp. 169-182, Springer-Verlag, 1985.
17. R. Rivest, A. Shamir, “Efficient Factoring Based on Partial Information”, In Advances in Cryptology (Eurocrypt ’85), Lecture Notes in Computer Science, Volume 219 , pp. 31-34, Springer-Verlag, 1986
18. R. Rivest, A. Shamir, L. Adleman, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, Communications of the ACM, Volume 21(2), pp. 120-126, 1978
19. P. Shor, “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”, Proceedings 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, IEEE Computer Science Press pp. 124-134, 1994
20. R. Steinfeld, Y. Zheng, “An Advantage of Low-Exponent RSA with Modulus Primes Sharing Least Significant Bits”, CT-RSA ’01, Lecture Notes in Computer Science Vol. 2020, pp. 52-62, 2001
21. S. A. Vanstone, R. J. Zuccherato,“Short RSA Keys and Their Generation”, Journal of Cryptology 8 number 2, pp. 101-114, 1995

22. A. Young, M. Yung, “The Prevalence of Kleptographic Attacks on Discrete-Log Based Cryptosystems”, Advances in Cryptology (Crypto ’97), Lecture Notes in Computer Science, Volume 1294, pp. 62-74, Springer-Verlag, 1997