The Security and Efficiency of Micciancio's Cryptosystem

Christoph Ludwig

FB 20, Technische Universität Darmstadt
Hochschulstr. 10, D-64289 Darmstadt, Germany
cludwig@cdc.informatik.tu-darmstadt.de

Abstract. We report experiments on the security of the GGH-like cryptosystem proposed by Micciancio. Based on these experiments, we conclude that the system can be securely used only in lattice dimensions ≥ 782. Further experiments on the efficiency of the system show that it requires key sizes of 1 MByte and more and that the key generation as well as the decryption take inacceptibly long. Therefore, Micciancio's cryptosystem seems currently far from being practical.
Keywords: Lattices, trapdoor functions, public-key encryption.

1 Introduction

Lattice theory has important applications in cryptography. Perhaps the most widely known are attacks against many knapsack based cryptosystems, RSA, and DSA. But lattice theory has also been used in security proofs and in the construction of new cryptosystems. In contrast to cryptosystems based on factoring or discrete logarithms, there are no known quantum algorithms that allow to attack lattice based cryptosystems significantly faster than with classical algorithms. Lattice based cryptography may therefore serve as a long term alternative to established cryptosystems.

In 2001, Micciancio [Mic01] proposed a cryptosystem strongly related to the GGH system [GGH97b] that is based on the closest vector problem, one of the classical problems in lattice theory. His system seems attractive not only because its base problem is cryptographically independent from established systems like RSA or elliptic curve cryptography. It also offers several advantages over GGH: The key and ciphertext sizes are asymptotically one order of magnitude smaller than in the GGH system. Most important, Micciancio's choice of public keys offers the highest security achievable with a cryptosystem of GGH type.

But there is no comprehensive analysis of the lattice dimension reqired for the system to be secure or of the system's efficiency in practice yet.

In this paper we determine the minimum lattice dimension that guarantees security for Micciancio's cryptosystem and we discuss the efficiency of Micciancio's system in that circumstances. Our result uses the methodology of Lenstra and Verheul [LV01] and is based on extensive experiments that we carried out. In particular, we found that secure instances must be in dimension $n \geq 782$, which is much larger than dimension 500 conjectured by Micciancio. Recently, Schnorr [Sch02] published novel algorithms for solving the closest vector problem. If we assume the improvements in the asymptotic running time take full effect in practice, then the required dimension jumps up to 2094.

Our further experiments on the space and time efficiency of the system show that one must expect public keys of at least 1 MByte. Without significant improvements in algorithms for computing the Hermite normal form, the key generation will take days. The encryption procedure is quite fast, but the decryption suffers from an instability of Babai's nearest plane algorithm when using floating point arithmetic. We show how to overcome this, but our experiments show a decryption time of over 73 minutes in dimension 800, not including the required orthogonalization of the private lattice basis. If the orthogonalization is precomputed, one needs to store in dimension 800 more than 4.8 MByte in addition to the private key.

We therefore have to conclude that currently Micciancio's system is far from being practical.

In section 2 we give some basic definitions and results from lattice theory and outline cryptosystems of GGH type in general as well as Micciancio's system in particular. In section 3 we describe possible attack strategies, report on our experiments and discuss the extrapolation of their results. Finally, in section 4 we determine the time and space efficiency of both the key size and key generation as well as the encryption and decryption functions. We also describe how to successfully decrypt ciphertexts with a floating point variant of Babai's algorithm.

2 Lattices and Micciancio's Cryptosystem

2.1 Preliminaries on Lattices

A k-dimensional lattice L is the $\mathbb{Z}$-span of some $\mathbb{R}$-linear independent *lattice basis* $B = \{b_1, \ldots, b_k\} \subset \mathbb{R}^n$, i. e. $L = \{\sum_{i=1}^{k} a_i b_i : a_1, \ldots, a_k \in \mathbb{Z}\}$. By abuse of notation, we identify the basis B with the $n \times k$ matrix $B = [b_1, \ldots, b_k]$. In the following, we only consider fully dimensional integral lattices, so $k = n$ and $B \in \mathbb{Z}^{n \times n}$. All vector norms and scalar products are Euclidean.

The skewedness of a basis B is measured by the *orthogonalization defect* $\mathrm{odef}(B) := \det(B)^{-1}\prod_{i=1}^{n}\|b_i\|$. By Hadamard's inequality, we have $\mathrm{odef}(B) \geq 1$ with equality if and only if B is an orthogonal basis. $B^* = [b_1^*, \ldots, b_n^*]$ denotes the Gram-Schmidt orthogonalization of B,

$$b_i^* = b_i - \sum_{j=1}^{i-1} \frac{\langle b_i, b_j^* \rangle}{\|b_j^*\|^2} b_j^* \quad \text{for } i = 1, \ldots, n\,.$$

The *height of* B, short $h(B)$, is the height of the cube spanned by the orthogonalized basis $[b_1^*, \ldots, b_n^*]$, i. e. $h(B) = \min\{\|b_1^*\|, \ldots, \|b_n^*\|\}$. (Note that $h(B)$ depends on the order of the basis vectors.)

The *Closest Vector Problem* (CVP) is a classical lattice problem: Given a vector $x \in \mathbb{R}^n$ and some basis B of a lattice L, find a lattice point $v \in L$ with minimal distance $\|x - v\|$. More than twenty years ago, van Emde Boas [EB81] showed CVP is NP-hard. A related problem is the *Shortest Vector Problem* (SVP) that asks for a nonzero vector of minimal length in L. SVP is NP-hard under randomized reductions [Ajt98, Mic98].

Among the infinitely many bases of a lattice, some are better suited to solve CVP instances than others. Generally speaking, the smaller the vectors in B and the less the orthogonalization defect $\mathrm{odef}(B)$, the easier are corresponding instances of CVP and SVP. Given an arbitrary lattice basis, it is hard to compute a sufficiently "good" basis of the same lattice. There are algorithms to improve a basis, though. The most famous one is the polynomial-time LLL-reduction [LLL82, SE94, SH95]. An LLL-reduced basis contains an approximation to a shortest lattice vector up to a factor exponential in the lattice dimension.

2.2 Cryptosystems of GGH Type

Cryptosystems of GGH type can be described as follows: One chooses a basis $R \in \mathbb{Z}^{n\times n}$ of some lattice L that allows to efficiently compute the closest lattice point for all vectors $x \in \mathbb{R}^n$ within some reasonable distance ρ from L. R serves as private key. The public key B is another basis of L that is a "bad" starting point for solving the CVP. The CVP in L has to be infeasible for everyone who knows only the basis B. Typically, $\mathrm{odef}(R)$ is very small while $\mathrm{odef}(B)$ is very big. The trapdoor function is

$$f_B : \mathbb{Z}^n \times \mathcal{E} \to \mathbb{Z}^n : (x, e) \mapsto Bx + e$$

where $\mathcal{E} \subset \{e \in \mathbb{Z}^n : \|e\| \leq \rho\}$ is a set of error vectors not longer than ρ. Given $c = f_B(x, e)$, everyone in possession of the trapdoor R (or some other sufficiently reduced basis) can efficiently compute the lattice vector $v = Bx \in L$ closest to c and therefore determine $e = c - v$ and $x = B^{-1}v$. On the other hand, whoever is able to recover x or e is also able to compute the lattice point v. By our requirement on B, this is infeasible for everyone who knows only the public key.

For the GGH challenges [GGH97a] in dimension 200 through 400, R was $n^{1/2}I + Q$ for some uniformly chosen $Q \in \{-4, \dots, 4\}^{n \times n}$. B was generated from R by repeated elementary column transformations. The message was encoded in x while e was chosen randomly in $\mathcal{E} = \{\pm 3\}^n$. This particular choice of $\mathcal{E}$ allowed Nguyen [Ngu99] to break the GGH challenges.

2.3 Micciancio's Cryptosystem

Micciancio observed in [Mic01] that not only the choice of $\mathcal{E}$ rendered GGH unnecessarily weak. Because of Babai's nearest-plane method for computing near lattice vectors [Bab86], every basis R of L with $2\rho < h(R)$ qualifies as trapdoor. There is no reason to restrict R to bases near the coordinate axes as GGH does. Micciancio proposed R to be the LLL-reduction of a matrix uniformly chosen in $\{-n, \dots, n\}^{n \times n}$ and $\mathcal{E} = \{e \in \mathbb{Z}^n : \|e\| < h(R)/2\}$.

Every lattice has a unique basis $H = (h_{i,j})$ in *Hermite normal form* (HNF), i. e., H is upper triangular, all diagonal elements $h_{i,i}$ are positive, and $0 \leq h_{i,j} < h_{i,i}$ for all $1 \leq i < j \leq n$. Since the HNF requires $O(n^2 \log n)$ bit space rather than the $O(n^3 \log n)$ bit of the public key in the original GGH proposal, it is attractive to choose H as public key. Micciancio pointed out that one does not forgo any potential security by this choice, because there are well known polynomial-time algorithms for computing the HNF basis H from any lattice basis. In particular, an attacker can compute H from any public key B.

If the message is encoded in the error vector e rather than in x, the size of the ciphertext $c = Hx + e$ can also be reduced. There is a unique $c' \in \mathbb{Z}^n$ such that $c' \equiv c \bmod L$ and $H^{-1}c' \in [0, 1[^n$. Then $e = c' - v'$ where v' is the lattice point closest to c'. c' can be easily computed from c as well as from e with knowledge of the public basis H only. c' requires only $O(n \log n)$ bits in contrast to $O(n^2 \log n)$ bits in the GGH proposal, because $0 \leq c'_i \leq \sum_{j=i}^{n} h_{i,j}$.

3 Security of Micciancio's Cryptosystem

There are two main strategies to solve CVP instances, Babai's nearest-plane method [Bab86] and the embedding method [Ngu99]. Recently, Schnorr published a sampling method [Sch02]. We do not have practical experience with his approach yet, but we take Schnorr's results into account when we extrapolate the running time of our experiments.

Babai's method has the advantage that it always returns the closest (rather than just a near) vector provided the error vector is shorter than half of the basis' height. Hence, it guarantees that the private basis R is indeed a trapdoor. But while the height of Micciancio's private bases grows linearly, the height of an LLL reduction of the HNF is negligible, see Fig. 1. An attacker is therefore more likely to succeed with the embedding method.

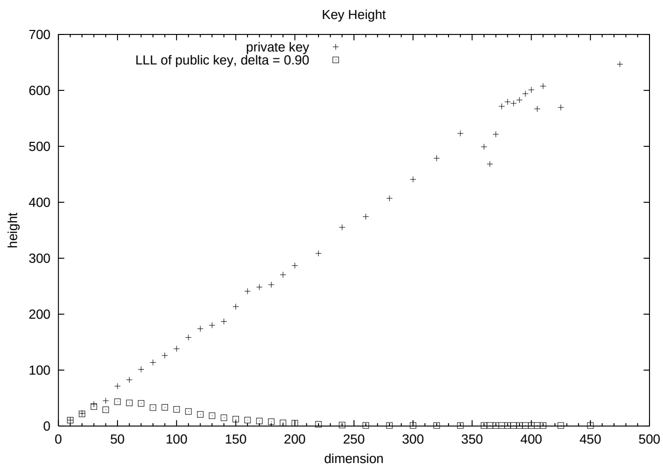

Fig. 1. Height of private and reduced public key.

The error vector in Micciancio's system is always shorter than the shortest nonzero vector in the lattice generated by H. The idea of the embedding method is therefore to transform the CVP into a SVP instance. One adds the ciphertext c to the public basis embedded in $\mathbb{R}^{n+1}$. If a lattice basis reduction actually yields the shortest vector in the lattice generated by the new basis, then the attacker recovered the error vector e up to the sign.

3.1 Attacks in Low Dimensions

An approximate solution to the underlying CVP or – after embedding – to the SVP is not sufficient to break Micciancio's cryptosystem. But a theoretical analysis of the polynomial time lattice reduction algorithms guarantees only an approximation up to a factor exponential in the lattice dimension. However, the LLL algorithm and its improved *Block Korkine-Zolotarev* (BKZ) variant often perform much better than one can expect from the theoretical bounds. It is therefore reasonable to attack Micciancio's system by BKZ reduction.

There is no comprehensive analysis or description of the behavior of lattice reduction algorithms in practice. Thus, it is somewhat unclear how an attacker is to choose the reduction parameters and heuristics in order to maximize the likeliness of success while minimizing the running time. In our experiments, we settled for three reductions. First we tried an LLL reduction with $\delta = 0.99$, then a pruningless BKZ reduction with $\delta = 0.99$ and block size $k = 20$. Finally, we tried a BKZ reduction with $\delta = 0.99$, $k = 60$ and pruning factor 20.

Because of the problems with Babai's nearest plane algorithm reported in section 4.2, one may prefer a different algorithm for decryption, e. g. the simple round-off method. But then the error vectors must be significantly shorter than $h(R)/2$. We therefore ran our experiments with error vectors of varying length.

We created key pairs for Micciancio's system up to dimension 280. For each key pair and $r = 10\%, 20\%, \ldots, 100\%$, we randomly chose up to 5 error vectors, each of length $r\rho$, and encrypted them as described in section (2.3). All in all, we generated more than 9800 ciphertexts. The results of our attacks on these ciphertexts are shown in Tabs. 1 and 2. All experiments were performed on Sun Blades 100. All machines have an 500 MHz UltraSparc IIE processor and 1 GByte RAM. For simplicity, we assume the performance of these machines to be 500 MIPS. The lattice reductions were done with the extended exponent double variant of the LLL and BKZ implementations with Givens orthogonalization in Shoup's NTL library [Sho01].

The results allow several observations: As long as the same algorithms are successful, the running time is independent from the length of the error vector. However, there is a relation between the error vector length and the likeliness that an attack with small block size succeeds. E, g., LLL recovered some messages even in dimension 280 if $r = 10\%$. But if $r = 100\%$, then LLL always failed in dimensions 180 and higher.

dim	relative length r of error vector in percent									
	10	20	30	40	50	60	70	80	90	100
50	100	100	100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100	100	100	100
110	100	100	100	100	100	100	100	100	100	100
120	100	100	100	100	100	100	100	100	94	92
130	100	100	100	100	100	100	90	100	92	92
140	100	100	100	100	100	94	88	100	88	62
150	100	100	100	100	100	94	90	62	68	60
160	100	100	100	100	100	90	64	40	48	27
180	100	98	100	98	78	48	40	8	12	16
190	100	100	100	89	71	39	26	38	0	30
200	100	100	100	64	32	20	12	12	20	58
220	100	92	44	40	8	0	32	32	48	92
240	100	65	10	20	10	41	50	90	100	100
260	100	0	0	0	33	73	93	90	100	100
280	50	0	10	30	40	90	100	80	100	100

Table 1. Success Ratio of Embedding Attacks in Percent

dim	relative length r of error vector in percent									
	10	20	30	40	50	60	70	80	90	100
50	1	1	1	1	1	1	1	1	1	1
100	52	51	52	51	51	51	51	51	51	51
110	80	80	80	80	80	79	79	79	79	80
120	112	112	112	112	112	112	112	112	113	115
130	171	171	171	171	171	171	170	171	176	170
140	233	232	233	233	233	235	226	232	231	239
150	362	362	362	362	362	354	353	352	360	350
160	515	515	516	515	515	510	522	540	529	504
180	762	767	762	768	800	804	761	748	1091	1124
190	1147	1145	1145	1146	1136	1169	1366	1355		1909
200	1380	1381	1381	1446	1554	1615	1615	2445	2506	19984
220	2223	2194	2322	2213	2833		4367	4250	4834	5319
240	3412	3424	3877	5594	5687	6306	8309	8799	10629	26624
260	5104				12847	12426	14092	13796	16813	21763
280	6391		11667	10792	16004	21094	22942	25233	26944	38177

Table 2. Average Running Time of Successful Embedding Attacks in Seconds

The high success ratio in the lower right corner of Tab. 1 is due to the BKZ reduction with blocksize 20. BKZ with blocksize 60 and pruning never recovered a message if LLL and BKZ with blocksize 20 failed to do so.

3.2 Lenstra-Verheul Extrapolation

BKZ reduction with blocksize $k = n$ will always yield a shortest lattice vector and therefore recover the plaintext. Since BKZ requires $O(n^3 k^{k+o(k)} + n^4)$ steps, $O(\exp(n))$ is an asymptotic upper bound for the running time of the attacks.

In practice, the worst case approximation factor $(k/3)^{n/k}$ of the BKZ algorithm is too pessimistic. The algorithm often finds a shortest vector even if k is significantly smaller than n. Since the cryptosystem is insecure if an attacker stands a non-negligible chance to recover a plaintext, we cannot exclude the possibility that the running time of the attacks may be subexponential in practice. We therefore also extrapolated our data with the hypothetical running time bound $O(\exp(n^{1/2}))$. This particular choice is of course somewhat arbitrary, but it serves to demonstrate that in practice Micciancio's system may require higher dimensional lattices than the lower bound we determined.

Schnorr [Sch02] shows under two assumptions that his new sampling method approximates the closest vector up to a factor at most $(k/6)^{n/2k}$ in $O(n^3(k/6)^{k/4} + n^4)$ time, improving the exponent by a factor 4. His asymptotically even faster birthday sampling algorithms seem impractical in dimension ≥ 800 since their space requirements grow exponential in k.

We extrapolated our data in four scenarios: First, we considered only attacks with the BKZ algorith as implemented in the NTL library and optimisticly assumed the running time $O(\exp(n^{1/2}))$. Second, we determined the running time with the worst case running time bound $O(\exp(n))$. Finally, we repeated these extrapolations assuming the asymptotic improvements in Schnorr's sampling will take full effect in practice.

The resulting extrapolation functions are given in Tab. 3. Based upon the number of MIPS-years that are infeasible today, in 10 years, and in 20 years according to Lenstra and Verheul, the extrapolation functions allow to compute the minimal lattice dimension in which Micciancio's system should be considered secure (Tab. 4). Even if we assume an exponential running time of the attacks and if we ignore potential improvements by Schnorr's sampling algorithms or other less drastic algorithmic improvements, we soon need lattice dimensions ≥ 800, significantly higher than

sampling	r in percent	$O(\exp(n))$	$O(\exp(n^{1/2}))$
no	10	$\exp(0.0349 * n - 10.9887)$	$\exp(0.8565 * n^{1/2} - 15.9714)$
	50	$\exp(0.0383 * n - 11.3600)$	$\exp(0.9307 * n^{1/2} - 16.7198)$
	100	$\exp(0.0469 * n - 12.4045)$	$\exp(1.1093 * n^{1/2} - 18.5544)$
yes	10	$\exp(0.0087 * n - 2.7472)$	$\exp(0.2141 * n^{1/2} - 3.9928)$
	50	$\exp(0.0096 * n - 2.8400)$	$\exp(0.2327 * n^{1/2} - 4.1800)$
	100	$\exp(0.0117 * n - 3.1011)$	$\exp(0.2773 * n^{1/2} - 4.6386)$

Table 3. Extrapolation of Running Time in MIPS-Year

conjectured by Micciancio. If we concede an attacker may employ improved algorithms or may not need to increase the block size linearly, then the required lattice dimension is even a multiple of Micciancio's conjecture.

If we chose a faster decryption procedure that requires shorter error vectors than the required lattice dimension was also significantly higher.

sampling	r in percent	year	$O(\exp(n))$	$O(\exp(n^{1/2}))$
no	10	2003	1010	2209
		2013	1162	2831
		2023	1314	3530
	50	2003	930	1941
		2013	1069	2477
		2023	1208	3078
	100	2003	782	1492
		2013	895	1885
		2023	1008	2323
yes	10	2003	3094	17436
		2013	3703	24608
		2023	4311	33009
	50	2003	2831	14965
		2013	3385	21076
		2023	3940	28229
	100	2003	2334	10875
		2013	2787	15240
		2023	3239	20338

Table 4. Required Lattice Dimensions

4 Efficiency of Micciancio's System

4.1 Key Size and Key Generation

We generated 260 key pairs in lattice dimensions up to 475. The private key generation was dominated by the LLL reduction of the randomly chosen bases, that took up to 58 minutes. For the computation of the public keys, we tested several HNF implementations in dimension 200 and decided to use the implementation of Kannan's algorithm in the LiDIA library [LG01] that ran for about 4 hours in dimension 475.

We also generated a single key pair in dimension 800. The private key was completed after 4.5 hours, the public key generation took about 46 hours.

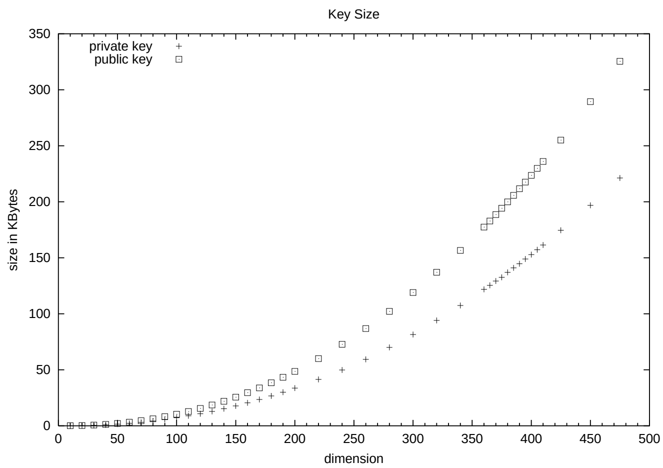

Fig. 2. Key Sizes

One of the advantages of Micciancio's proposals over the original GGH scheme is that it reduces the asymptotic public key size from $O(n^3 \log n)$ to $O(n^2 \log n)$. Nevertheless, he found that the HNF basis of the GGH challenge in dimension 400 still requires 140 KByte. In our experiments, the public keys turned out to be even bigger due to the different choice of private bases. Ignoring any bookkeeping overhead and accumulating the bitlength of all nonzero entries, we found the public keys require 1.0 MByte in dimension 800, for lower dimensions see Fig. 2. The private keys are consistently smaller by a factor 0.69.

4.2 Encryption and Decryption

Algorithm 1 Babai's nearest-plane algorithm

Require: $R = [r_1, \dots, r_n] \in \mathbb{Z}^{n \times n}$ is basis of lattice L.
$R^* = [r_1^*, \dots, r_n^*] \in \mathbb{Q}^{n \times n}$ is Gram-Schmidt orthogonalization of R.
$c \in \mathbb{R}^n$
Ensure: $v \in L$ is lattice point close to c.
1: **for** $i = n$ **downto** 1 **do**
2: $\quad x_i \leftarrow \lceil \langle c, r_i^* \rangle / \|r_i^*\|^2 \rfloor$ /* round to closest integer */
3: $\quad c \leftarrow c - x_i r_i$
4: **end for**
5: return $v \leftarrow R(x_1, \dots, x_n)^t$

Micciancio's encryption procedure simply reduces the message encoded in the error vector e modulo the public basis H as described in section 2.3. This involves solving one linear system and computing one matrix vector product. Since H is upper triangular and typically sparse, both operations are relatively cheap. The encryption of a message in dimension 800 takes on the Sun Blades about 0.29 seconds.

The situation is drastically different with the decryption procedure. Assuming exact arithmetic and $2\rho < h(R)$, Babai's nearest-plane algorithm correctly decrypts all ciphertexts. But even small errors in the orthogonalized basis R^* cause the algorithm to fail in line 2 of Alg. 1. In our experiments, we had to compute the orthogonalization with a precision of several thousand bits in order to be reliably able to decrypt messages with an unmodified implementation of Babai's algorithm. The running time and space requirements render this approach inacceptable.

Two observations nevertheless allow to decrypt ciphertexts c in practice: The quality of the approximation to the closest vector computed by Babai's algorithm depends on the length $\|c\|$. And even if it can't determine the closest vector due to rounding errors, the algorithm is likely to find a more or less close lattice vector v'. I. e., $c' := c - v'$ is shorter than c. (Otherwise, we can't avoid to restart the algorithm with higher precision. But in our experiments, we never encountered this case.) If $2\|c'\| < h(R)$, then we know $c' = e$ and we terminate. Otherwise we replace c by c' and apply Babai's algorithm again. Since c' is shorter than c, the algorithm returns a better approximation than in the previous iteration. c' will eventually be short enough to compute the closest vector.

It turned out to be most efficient to compute R^* with hardware floating point arithmetic, i. e. with 53 bit precision. For the actual orthogo-

Dimension	Precision	Orthogonalization	Decryption	# iterations (double / xdouble)
200	53	0.6	0.3	22 / 22
	200	1.3	0.7	41
	500	4.2	4.5	4
	2000	36.4	15.1	1
400	53	4.7	4.3	23 / 79
	250	11.7	6.2	19
	500	31.1	13.4	10
800	53	40.1	73.7	23 / 215

Table 5. Performance of Decryption (in Minutes).

nalization, we preferred the much more stable QR decomposition with Fast Givens Rotations [GvL96] over the modified Gram-Schmidt procedure. Due to an exponent overflow, only about the last 23 iterations can use a hardware arithmetic implementation of Babai's algorithm, though. Decryption of a random ciphertext in dimension 800 requires about 240 iterations. Each iteration decreases the length of c by about 13 decimal places.

Tab. 5 shows the running time for computing the QR decomposition as well for the actual decryption procedure with varying floating point precisions. If the precision was 53 bit, then we used the hardware datatype *double* or – if necessary within the nearest-plane algorithm – the extended exponent datatype *xdouble* from NTL. If the precision was more than 53 bit, than all computations were done with NTL's floating point type *RR*. The decryption of a message, excluding the orthogonalization, takes more than 73 minutes in dimension 800.

Since the orthogonalization of the private basis takes additional 40 minutes in dimension 800, it is attractive to precompute the orthogonalized basis. However, even with only 53 bit precision, the orthogonalized basis requires more than 4.8 MByte memory.

5 Conclusion

Much higher lattice dimensions are necessary for Micciancio's cryptosystem to yield a secure cryptosystem than previously conjectured. Considering recent algorithmic progress, the required dimension may very well be > 1000. In consequence, the keys require 1 MByte space and more, the key generation as suggested by Micciancio takes days and the decryption with an iterated nearest-plane algorithm takes at least minutes.

For all practical purposes, Micciancio's cryptosystem seems therefore currently not viable.

References

[Ajt98] Miklós Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 10–19. ACM Press, 1998.

[Bab86] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[EB81] P. van Emde Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. Technical Report 81-04, University of Amsterdam, Department of Mathematics, Netherlands, 1981.

[GGH97a] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. The Goldreich-Goldwasser-Halevi cryptosystem, challenge page. URL `http://theory.lcs.mit.edu/cis/lattice/challenge.html`, 1997.

[GGH97b] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski, Jr., editor, *Advances in Cryptology – Crypto'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 112 – 131. Springer-Verlag, 1997.

[GvL96] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[LG01] LiDIA-Group. LiDIA – A library for computational number theory. TU Darmstadt, 2001. Release 2.1pre5.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[LV01] Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, 2001.

[Mic98] Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. In *IEEE Symposium on Foundations of Computer Science*, pages 92–98, 1998.

[Mic01] Daniele Micciancio. Improving lattice based cryptosystems using the Hermite normal form. In Silverman [Sil01], pages 126–145.

[Ngu99] Phong Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto'97. In M. Wiener, editor, *Advances in Cryptology – Crypto'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 288–304. Springer-Verlag, 1999.

[Sch02] Claus Peter Schnorr. Lattice reduction by random sampling and birthday methods. available at `http://www.mi.informatik.uni-frankfurt.de/research/papers.html`, 2002. Preprint.

[SE94] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Programming*, 66:181–199, 1994.

[SH95] C. P. Schnorr and H. H. Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *Advances in Cryptology – Eurocrypt'95*, volume 921 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 1995.

[Sho01] Victor Shoup. NTL – a library for doing number theory. URL `http://www.shoup.net/ntl/index.html`, 2001. Release 5.2.

[Sil01] Joseph H. Silverman, editor. *Cryptography and Lattices*, volume 2146 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.