SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

The GGH Encryption Scheme – A Lattice-Based Cryptosystem

av

Amelie Schenström

2016 - No 11

The GGH Encryption Scheme – A Lattice-Based Cryptosystem

Amelie Schenström

Självständigt arbete i matematik 15 högskolepoäng, grundnivå

Handledare: Jonas Bergström

2016

Abstract. The GGH encryption system, which is a cryptosystem based upon the mathematical theory of lattices, was proposed in 1997. Only two years after it was published, great flaws were found in the scheme making it unsecure in the dimensions proposed. With a higher dimension the scheme would be impractical and thus it was considered to be dead. However, improvements have been made since then. We will explore the properties of the GGH encryption scheme and the ones of a proposed improvement to see if it is an encryption system that can be sufficiently secure and of practical use.

Contents

1 Introduction

The ability to send information without the wrong people reading it has been a highly regarded skill for a long time. But during the second world war it was more important than ever before. Every message was sent through an unsecure channel and the enemy was listening in on every message sent. The security of the messages encrypted had a crucial impact on the warfare. The ability to listen on the enemy's conversations played a significant role in winning the war.

Today we use cryptography in many areas of life. It is not only used to give military orders or to have a secure conversation between people in an unsecure channel. It is now used to a vast extent. But the cryptography today is very different from the ones used earlier. A large part of the cryptography we use today is less technical and more mathematical. These cryptographic applications are based on mathematical problems that are considered to be hard to solve. A typical, well used, cryptosystem of this type is RSA. It is based on the hard mathematical problem of factorizing a large integer. But what would happen if we could find a fast way to solve this kind of problem? What if we all of a sudden could easily solve all the mathematical problems underlying the cryptosystems of today? Then we would have no safe way to transfer delicate information between remote parties.

This is something that may happen in an not too distant future. The quantum computers are known to be able to solve integer factorization fast and making the numbers larger will not make for a sufficiently fast encryption method. So when quantum computers are at use, all systems used today to secure our data will not be secure at all.

We are going to need new types of cryptosystems for transfering information securily. There are cryptosystems that are considered to be post-quantum cryptosytems which means that they will be secure against attacks performed by a quantum computer. This field of cryptography is fairly young and there have not been so many suggestions for cryptosystems of this type intended to be used in practice. One type of post-quantum cryptography are the lattice-based ones which are the focus of this text. Some of the lattice-based cryptosystems seem to be resistant to both the type of attack used today and of the attacks of quantum computers. This property would make it secure for all types of attacks that are known.

Uptill today a few lattice-based cryptosystems have been proposed. There are systems such as GGH, NTRU and Ring-learning with errors. In this

text we will consider in the GGH encryption system which was proposed by Goldreich, Goldwasser and Halevi in 1997. Only two years after, in 1999, Nguyen showed that the scheme was not secure enough. He showed that every ciphertext leaks information about the plaintext and that the problem of decryption can be simplified. Does this mean that the scheme is dead? If so, can it be revived if the right improvements are made?

Several proposed improvements have been made since 1997, all of them using different techniques. We will look at one of them, the improvement proposed by Micciancio in 2001. His idea is, to make the scheme easier to implement one needs to use deterministic methods to choose keys and parameters instead of randomized ones. But at the same time he wants it to be more secure and tries to make a trapdoor function which includes both a higher level of security and deterministic choices of keys and parameters.

2 Basic Definitions and Properties of Lattices

Before setting up a lattice-based cryptosystem we need some definitions and properties. A lattice is similar to a vector space but instead of having coefficients that are real numbers the coefficients are integers.

Definition: Given a set $B = \{b_1, ..., b_n\}$ of n linearly independent vectors in $\mathbb{R}^n$, we define the lattice spanned by B as the set of all linear combinations of the b_i's with integer coefficients.

$$L(B) = \{\sum_i x_i b_i | x_i \in \mathbb{Z} \text{ for all i}\}$$

We call B a basis of the lattice $L(B)$ and any set of linearly independent vectors that spans L is basis for L. When a basis spans a lattice we also say that it generates that lattice.

Definition: A vector that belongs to the lattice L we call a lattice point or a lattice vector.

Definition: Let L be the lattice of n and let $b_1, b_2, ..., b_n$ be a basis for L. The fundamental domain (or the fundamental parallelepiped) for L corresponding to this basis is the following set

$$P(b_1, ...b_n) = \{a_1b_1 + a_2b_2 + ... + a_nb_n | 0 \leq a_i < 1, i = 1, 2, ..., n\}$$

For every vector $w \in \mathbb{R}^n$ there is a unique vector $v \in L$ and a unique vector $t \in P(B)$ such that $w = v + t$. We also say that a orthogonalized parallelepiped is the fundamental domain of the Gram-Schmidt reduced basis (see section 2.3), i.e. of a basis where every vector is orthogonal to all the other vectors. Notice that this basis, in general, will not span the lattice.

You can transform a basis of a lattice to some other basis of the same lattice using unimodular matrices.

Definition: A unimodular matrix is a square matrix with integer coefficients and with determinant equal to 1 or -1.

The following proposition is easy to prove.

Proposition: Two different bases for a lattice L are related by a unimodular matrix.

This means that the absolute value of the determinant for each basis of a lattice L is the same. And given any basis B and any unimodular matrix U, $B \cdot U$ gives us a new basis for the lattice. A basis of a lattice is not unique but a basis in Hermite normal form is uniquely determined.

Definition: A basis B, with integer coefficients, is in Hermite normal form if it is upper triangular, the elements of the diagonal are strictly positive, and for all other elements $b_{i,j}$, we have $0 \leq b_{i,j} < b_{i,i}$.

Proposition: To each lattice there is a unique basis in the Hermite normal form [6]. We denote it HNF(B).

Some properties of a lattice may effect how secure the cryposystem is. One of these properties is the gap of a lattice because if a lattice has a large gap the reduction algorithms (see section 2.3) will finish in less time.

Definition: The gap of a lattice is the ratio between the length of a second shortest vector and the length of a shortest non-zero vector.

We will talk about a "good basis" and a "bad basis". A good basis is very close to being orthogonal. We can measure how close a basis is to being orthogonal using the following ratio.

Definition: The Hadamard ratio of the basis B is $H(B) = \left(\frac{detB}{||b_1||||b_2||...||b_n||}\right)^{1/n}$. We have that $0 < H(B) \leq 1$, the closer this value is to one the more orthogonal is the basis. We will also call this the orthogonal defect.

2.1 Hard Lattice problems

The security of lattice-based cryptography is based on fundamental computational problems such as finding a shortest non-zero vector in the lattice or finding a vector in the lattice that is closest to a given vector not in the lattice. These problems are considered to be hard to compute if one is not given a good basis.

Definition: The shortest vector problem (SVP) consists of finding a non-zero vector $v \in L$, where L is a lattice, such that that minimizes the Euclidean norm $||v||$.

Definition: The closest vector problem (CVP) consists of finding a vector $v \in L$, where L is a lattice, that is closest to a given vector w not in L. This is the vector $v \in L$ minimizing the Euclidean norm $||w - v||$.

These are two of the most important hard computational problems regarding lattices, and they are the ones we are interested in.

2.2 Babai's Closest Vertex Algortihm

To solve a closest vector problem, one needs an algorithm. Babai's algorithm works in the case that the basis of the lattice is orthogonal enough. If we assume that it is, the algorithm works as follows:

Babai's Closest Vertex Algorithm: *Let $L \in \mathbb{R}^n$ be a lattice with basis $v_1, ..., v_n$, and let $w \in \mathbb{R}^n$ be an arbitrary vector. If the vectors in the basis are (sufficiently) orthogonal to one another, then the following algorithm solves CVP.*

Write $w = t_1v_1 + t_2v_2 + ... + t_nv_n$ with $t_1, ..., t_n \in \mathbb{R}$
Set $a_i = \lfloor t_i \rceil$ for $i = 1, ..., n$
Return the vector $v = a_1v_1 + a_2v_2 + ... + a_nv_n$

Where $\lfloor c \rceil$ means that we round c to the closest integer.

2.3 Reduction Algorithms

Say that we are given a basis B that is far from being a good basis in terms of orthogonality. To solve one of the hard lattice problems a good basis is needed, therefore we want to transform this basis into a better one. In a vector space one might use the Gram-Schmidt Algorithm to transform a basis V to an orthogonal basis V^*. However this does not work for lattices since the new basis V^*, most likely, will not have integer coefficients. For lattices this can instead be done by lattice reduction. The lattice reduction algorithms given below are based on the Gram-Schmidt algorithm, so let us start by recollect that algorithm.

Gram-Schmidt Algorithm: *Let $v_1, ..., v_n$ be a basis for a vector space $V \subset \mathbb{R}^m$. The algorithm below creates an orthogonal basis $v_1^*, ..., v_n^*$ for V.*

Set $v_1^* = v_1$
Loop i=2,3,...,n.
Compute $\mu_{ij} = v_i \cdot v_j^* / ||v_j^*||^2$ for $1 \leq j < i$
Set $v_i^* = v_i - \sum_{j=1}^{i-1} \mu_{ij} v_j^*$.
End Loop

One reduction algorithm is the LLL Reduction Algorithm [5]. We say that a basis $B = \{b_1, ..., b_n\}$ is LLL reduced if the following conditions are satisfied.

$$\textbf{Size condition: } |\mu_{i,j}| = \frac{|b_i \cdot b_j^*|}{||b_j^*||^2} \leq \frac{1}{2} \text{ for all } 1 \leq j < i \leq n$$

$$\textbf{Lovász Condition: } ||b_i^*||^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)||b_{i-1}||^2 \text{ for all } 1 < i \leq n$$

Where $B^* = \{b_1^*, ..., b_n^*\}$ is the associated Gram-Schmidt orthogonal basis.

The LLL Reduction Algorithm: *Let $\{b_1, ..., b_n\}$ be a basis of a lattice L contained in $\mathbb{Z}^n$. The algorithm given below returns a LLL reduced basis for L.*

Input a basis $\{b_1, ...b_n\}$ for a lattice L
Set k=2
Set $b_1^* = b_1$
Loop while $k \leq n$
Loop Down j=k-1,k-2,...,2,1
Set $b_k = b_k - \lfloor \mu_{k,j} \rceil b_j$
End j Loop
If $||b_k^*||^2 \geq (\frac{3}{4} - \mu_{k,k-1}^2)||b_{k-1}^*||^2$
Set k=k+1
Else
Swap b_{k-1} and b_k
Set $k = max(k-1, 2)$
End If
End k Loop
Return LLL reduced basis $\{b_1, ..., b_n\}$

where $\{b_1^*, ..., b_n^*\}$ is the Gram-Schmidt orthogonal basis.

Another lattice reduction algorithm is the BKZ reduction algorithm which is based on KZ-reduction (or Korkin-Zolotarev reduction). First we define a map

$$\pi : L \to \mathbb{R}^n,$$

$$\pi_i(b) = b - \sum_{j=1}^{i} \frac{b \cdot b_j^*}{||b_j^*||^2} b_j^*.$$

Where any $B^* = \{b_1^*, b_2^*, ...b_n^*\}$ is the Gram-Schmidt orthogonalized vectors of $B = \{b_1, b_2, ...\}$. Now we define what it means to be KZ-reduced.

Definition: A basis $b_1, ..., b_n$ for a lattice L is called KZ reduced if it satisfies the conditions:

1. b_1 is the shortest non-zero vector in L.
2. For i=2,3,...,n, the vector b_i is chosen such that $\pi_{i-1}(b_i)$ is the shortest non-zero vector in $\pi_{i-1}(L)$.
3. For al $1 \leq i < j \leq n$, we have $|\pi_{i-1}(b_i) \cdot \pi_{i-1}(b_j)| \leq \frac{1}{2}||\pi_{i-1}(v_i)||^2$.

A basis that has been reduced by KZ-reduction is in general much better than a LLL-reduced basis. The first vector in a basis that is KZ-reduced is a solution to the SVP.

There is also a *block Korkin-Zolotarev* [8] version of the LLL algorithm. It replaces the swap step in the LLL algorithm by a block reduction step. In BKZ you work with blocks of vectors of length β

$$b_k, b_{k+1}, ..., b_{k+\beta-1}$$

and these are replaced with the KZ-reduced basis the spans that same sub-lattice.

2.4 The Embedding Technique

The embedding technique is a way to solve the CVP by defining a new lattice and then using LLL-reduction to find the smallest vector of that lattice [2].

We have the basis B of the lattice and a vector $w \in \mathbb{R}^n$ that is not a lattice point. A solution to the closest vector problem is the vector of integers $\{l_1, l_2, ..., l_n\}$ if

$$w \approx \sum_{i=1}^{n} l_i b_i$$

If we put $e = w - \sum_{i=1}^{n} l_i b_i$ then $||e||$ is small. Now we define a lattice L^* that includes both the basis B and the short vector w. We define the lattice L^* by the matrix

$$B^* = \begin{pmatrix} b_1 & 0 \\ b_2 & 0 \\ . & \\ . & \\ . & \\ b_n & 0 \\ w & 1 \end{pmatrix}$$

We reduce this basis and since w is e plus a linear combination of the basis B, the reduction will result in one of the vectors being e. If e is small enough then finding e is a shortest vector problem. Which means that we might be able to find e by solving the SVP of the lattice L^*. Then to solve the CVP one subtracts e from w.

As far as the vector w is close enough to a lattice point the closest vector problem can be reduced to a shortest vector problem using the technique described.

3 GGH Encryption Scheme

The GGH cryptosystem is the lattice-based cryptosystem introduced in 1997 by Goldreich, Goldwasser and Halevi [3] which is based on the difficulty to reduce lattices.

Let us first set a security parameter to be (n, σ), where n is the dimension of the lattice space and σ is the parameter determining the size of the error vector. We will start with a lattice Ł $\in \mathbb{Z}^n$ which is defined by a matrix R, which is a reduced basis i.e. a basis R where $H(R)$ is close to one. This basis R will be our private key, private basis. You can read about the way to choose a matrix R in [3].

Now we want to generate a public basis B. This public basis is obtained from the private one. The public basis will not be a reduced basis, that is $H(B)$ is not close to one. We can obtain B from R by using many "mixing" steps, we take one basis vector and add a random integer linear combination of the other vectors to it. Or R can be transformed into B by multiplying R with some "random" unimodular matrices.

We choose a message $m \in \mathbb{Z}^n$ that is encrypted into $c = mB + e$. Where e is the error vector chosen uniformly from $\{-\sigma, \sigma\}^n$. Other methods to encrypt is discussed in the article but this is the recognized encryption method.

The larger the parameter σ is, the harder the CVP is expected to be. But for large σ the decryption process might not succeed.

To obtain the plaintext from the ciphertext we use Babai's closest vertex algorithm to first find the vector $v \in L$ that is closest to c. Then we compute $B^{-1}v$ to find m.

4 Cryptanalysis of the GGH Cryptosystem

In 1999 Phong Nguyen claimed to have found security issues in the GGH cryptosystem [7]. He claims that every ciphertext leaks information about the plaintext.

4.1 Leaking Remainders

If (n, σ) are the sequrity parameter and B a public basis then the plaintext is encrypted as

$$c = mB + e \tag{1}$$

where $m \in \mathbb{Z}$ is the plaintext being encrypted, $c \in \mathbb{Z}$ is the ciphertext and $e \in \{\pm\sigma\}$ is the error vector. Nguyen claims that the equation of encryption has a flaw. By an appropriate choice of integer and (1) modulo that integer will make the error vector e disappear. This would give us m modulo the chosen integer. Every entry of e is either σ or $-\sigma$ so the natural choice of integer would be σ but instead he chooses 2σ. Letting $s = (\sigma, ..., \sigma)$ gives $e + s \equiv 0 \pmod{2\sigma}$ so that

$$c + s \equiv mB \pmod{2\sigma} \tag{2}$$

Solving this system gives m modulo 2σ. The questions now are: how many solutions are there, and how do we compute them?

The problem of solving a linear system $y = xB$ (mod N) where the vector y, the matrix B and the modulus N are all known, has at least one solution. Two solutions will differ by an element of the kernel of B, which equals

$\{x \in \mathbb{Z}^n | cB \equiv 0 \text{ (mod N)}\}$. This means that if we find one solution to the equation then the rest can be found from the kernel of B. The number of solutions is equal to the cardinal of the kernel. The simplest case of solving the linear system is if B is invertible modulo N, this happens if $det(B)$ is coprime to N and then there is only one solution. The solution is found by matrix inversion $x = yb^{-1}$ (mod N).

Nguyen shows that with a considerable probability, the public basis B is invertible modulo 2σ. This gives any plaintext modulo 2σ.

When the matrix is not invertible then Nguyen shows that the kernel is usually very small. He starts with discussing the case of a prime modulus. Then the kernel is a $\mathbb{Z}_p$-vector space and if the dimension of the kernel is d, then the number of solutions is p^d. It is clear that if both p and d are small then the number of solutions is small. He finds that with a high probability the kernel has a dimension less than 2. This means that the solutions tho the modular system is at most p^2. And he claims that the solutions are easy to compute.

If the modulus N is not a prime but is square-free, $N = p_1...p_s$ where $p_i \neq p_j$ for $i \neq j$, then the solutions can be found by using the Chinese Remainders from each solution modulo p_i. The total number of solutions is found by multiplying the number of solutions for each prime. For each prime one can also find the proportion of a matrix with respect to its kernel using the same argument as previously stated. He finds that only a very small part of the matrices modulo 6 (which is two times the suggested parameter $\sigma = 3$) have a kernel with more than 12 elements.

If N is not square-free the methods used before does not apply, but the solutions may be obtained. But this case is not relevant for the suggested parameters.

This means that for the suggested choice of parameters (n, σ) and for any ciphertext c, the linear system, most likely, has very few solutions. Because of the fact that the public basis B had a small kernel with high probability. Even though the encryption scheme is probabilistic, one can check whether a plaintext corresponds to a given ciphertext without knowing all of the plaintext. One can also check whether two ciphertext correspond to the same plaintext, with high probability.

4.2 Simplifying the Closest Vector Problem

Say that we have found the plaintext m modulo 2σ, and denote this by $m_{2\sigma}$. Knowing this will simplify the decryption problem which is based on the CVP. The encryption function is

$$c = mB + e.$$

Using the fact that we know $m_{2\sigma}$ we get that

$$c - m_{2\sigma}B = (m - m_{2\sigma})B + e$$

But the vector $m - m_{2\sigma}$ is on the form $2\sigma m'$, $m' \in \mathbb{Z}^n$. Therefore,

$$\frac{c - m_{2\sigma}B}{2\sigma} = m'B + \frac{e}{2\sigma}.$$

The left-hand side is known so the equation above is a closest vector problem where the error vector, $\frac{e}{2\sigma} \in \{\pm\frac{1}{2}\}^n$ which is smaller than previously. If one can solve this new CVP then one can easily solve the former one. This means that the problem of decryption has been reduced to an easier CVP.

From what we saw earlier, we do not always find $m_{2\sigma}$, but with high probability one can find it.

4.3 Repairing the Scheme and Conclusion

The way we choose the error vectors, always makes them shorter than the vectors of the lattice. This results in a gap of the embedded lattice, when the embedding attack is used (see section 2.4). If the gap is large then it is easier to reduce the lattice because a large gap result in a BKZ-reduction with a lower dimension and thus a faster reduction of the public basis.

There is another problem with having the error vectors in this specific form. We do not want an error which we know the value of modulo some integer that is chosen well, as we saw above. The most apparent way to avoid this is to choose the entries of the error vector e at random in the interval of $\big[-\sigma, \sigma\big]$. This result in a vector e with an approximate length that is

smaller than the original choice of error vector. A larger error can be found by choosing the entries of e randomly in $\{\pm\sigma, \pm(\sigma - 1)\}$, although this may result in a dangerous special form of the error vector.

The conclusion is that the special form of the error in the GGH scheme is dangerous since part of the information about the plaintext may be recovered. And the decryption problem can be reduced to a CVP considerably easier than the general CVP.

5 Improving GGH Using the Hermite Normal Form

In 2001 Daniele Micciancio published an improvement of the scheme of Goldwasser, Goldreich and Halevi [6]. His new cryptosystem is based on the public basis being the basis of the Hermite normal form.

Let $c = xB + e$ be the encryption function. In [3]two different encoding methods are considered, the first one is the one we have considered above, the message is encoded in the coefficients x, and the error vector is chosen randomly. The second method, the one that Micciancio considers, is that the message is encoded in the error vector, and instead we have x chosen at random. But he writes that his methods can be adapted to suit the first method as well.

A new trapdoor function is needed and has to be able to answer the questions: How is the private basis R chosen? How do we obtain the public basis B from R? How is the random vector x chosen? How do we choose the error vector e?

As was suggested in [3] we choose R of the following form $R = \lfloor\sqrt{n}\rceil \cdot I + Q$, I is the identity matrix and Q is a random perturbation matrix with entries in $\{-4, ..., +4\}$. R will be close to orthogonal since ± 4 is much smaller than $\sqrt{n}$. The way we obtain the public basis B, the random lattice vector x and the error vector e will be different than in the original scheme.

5.1 An Optimal GGH-like Trapdoor Function

We want to define a trapdoor function that works better than the GGH trapdoor function. Since it is hard to obtain random vectors and bases, Mic-

ciancio's idea is to replace the random choices for B and x with deterministic ones. The ones he proposes can be proven to be optimal from a security view point. We know how to obtain R, we let ρ be a correction radius. This means that if we use R we can correct any error that is smaller than ρ, as an example we have $\rho = \frac{1}{2} min_i ||r_i^*||$. The message will be encoded into the error vector, but we can only find the lattice point closest to e if it is inside the correction radius, i.e. the length of e is less than ρ.

5.1.1 Reducing Vectors Modulo a Basis

A lattice L defines an equivalence relation over $\mathbb{Z}^n$ in the following way: $v \equiv_L w$ if and only if $v - w \in L$. Starting with a basis B and for every vector $v \in \mathbb{Z}^n$ we have that $v = xB + r \equiv_L yB + r$ where $r \in \mathbb{Z}^n$. This can be written uniquely for any fundamental parallelepiped. We have that $v = xB + r$ where $r \in P(B)$, x and r is uniquely determined. In particular we have that for every point $v \in \mathbb{Z}^n$ there exists a unique point w in the orthogonalized parallelepiped $P(B^*) = \{\sum_i x_i b_i^* | 0 \leq x_i < 1\}$ such that v is congruent to w modulo L. This can be shown by induction on the dimension where the base case is the two-dimensional lattice. The orthogonal basis for this case is $B^* = \{b_1, b_2 - \frac{b_1 \cdot b_2}{||b_1||^2} b_1\}$. If $\frac{b_1 \cdot b_2}{||b_1||^2} = 1$ then the orthogonal parallelepiped is the rectangle with corners at the origin, v_1, $v_2 - v_1$ and v_2, then it is clear that there are no other lattice points in this parallelepiped and thus there is unique point w which is congruent to v modulo L. If instead $\frac{b_1 \cdot b_2}{||b_1||^2} < 1$ then the orgin and b_1 will still be corners but b_2 will lie on the side of the rectangle that is opposite the vector v_1 and no other lattice points is in the rectangle, then it is also clear that the point w is uniquely determined. For the last case $\frac{b_1 \cdot b_2}{||b_1||^2} > 1$, the origin and b_1 is two of the corners and the point $b_2 - b_1$ is on the opposite side to the vector b_1 and no other point is in the rectangle. It is therefore uniquely determined for this case as well. One can prove, with a similar argument, that if it holds for a $k-1$-dimensional space then it also holds for a k-dimensional case.

The unique element of $P(B^*)$ that is congruent to v modulo L we denote with v modulo B.

The definition of the reduced vector, v mod B, depends on the basis B but the equivalence relation $v \equiv_L w$ is not dependent on the basis.

If B is in Hermite normal form then $w = v$ mod B is an integer vector with the property $0 \leq w_i < b_{i,i}$. One can see this in the following way: if there would be some w_k not satisfying the inequality, then $w_k \geq b_{k,k}$ or $w_k < 0$ but

$0 < b^*_{k,k} \leq b_{k,k}$. This would mean that $w_k \notin P(B^*)$ and hence not reduced modulo B.

5.1.2 Choosing the Public Basis

The private basis R that we start from is a really good basis that makes it possible to solve the closest vector in the lattice. We want to transform this basis into the public basis without leaking too much information about the private basis. Instead of computing B by adding randomized transformations to R, we choose the public basis B of this new scheme to be the basis of the Hermite normal form, $B = \mathrm{HNF}(R)$ of R. Note that the private basis is a matrix with integer coefficients.

The basis of the Hermite normal form only depends on the lattice $L(R)$ generated by R and not on the particular basis used. This means that the public key leaks no information about the private key.

5.1.3 Adding a "Random" Lattice Point

Now we want to add a "random" vector xB of the lattice L to the error vector e. The optimal way to do this would be to choose xB uniformly. However this is not possible in practice. We will attain the same result by mapping the error e to its equivalence class $\left[e\right]_L$ since e is equivalent to $e - xB$ for some $x \in L$. We can use the reduced vector $e \bmod B$ as a representative for the class. So instead of adding a random lattice point to the error vector we reduce e modulo B, the private basis. This will give us the ciphertext $c = \left[e\right]_L \in P(B^*)$. Our trapdoor function is:

$$f(e) = e - xB = e \bmod B$$

remember that $B = \mathrm{HNF}(R)$, the Hermite normal form of the private basis R. The form of B makes the reduction modulo B very simple, thus the triangular form of B makes the trapdoor function simple.

5.1.4 The Trapdoor Function

Now all that is defined above is put together to a new trapdoor function. We choose R to be a private basis such that $\rho = \frac{1}{2} min_i ||r^*_i||$ is relatively large. As we have seen before the public basis is the Hermite normal form

of the basis, $B = \mathrm{HNF}(R)$. The public basis defines a function with the set of vectors with length smaller than ρ as the domain. When we apply the function to the error vector, it results in a point $f(e)$ in the orthogonalized parallelepiped $P(B^*)$ which is congruent to e modulo the lattice.

The error vector e is always close to the orgin, but the result of reducing it modulo B is possibly closer to some other lattice vector since the encryption function is $f(e) = e - xB \equiv e \bmod B$. Decrypting involves finding the closest lattice point to $f(e)$. This should not be possible using only the public basis B. One can compute the lattice point closest to $f(e)$ using the good private basis R as long as the length of e is smaller than ρ.

5.2 Analysis

Micciancio discusses the difference of security, space efficiency and running time compared to the orginal scheme. And first he proves that his new scheme is at least as secure as the original GGH encryption system.

He estimates the key size and the size of the ciphertext of both GGH and the modified scheme to compare them. The size of the key and ciphertext is significantly smaller for the new scheme than for the original GGH scheme.

When it comes to running time he claims that the encryption time is mainly dependent on the size of the public key. Since the key size is much smaller than the key size in GGH the encryption time will be much faster. The key generation is also much faster because Hermite normal form computations are generally fast but the old version based on applying LLL upon a matrix of high dimension is not. The decryption is the critical part since it is similar to the one of the original scheme. In high dimensions, the decryption can take several minutes. But he argues that since the decryption is strongly based on the choice of private basis, finding a way to generate such a basis might be more important.

5.3 Discussion and Conclusion

The trapdoor function that has been defined is at least as hard to break as the GGH encryption system. Although the original scheme was randomized and the new one is deterministic, it is still at least as secure. This means that GGH cannot be semantically secure since being so requires the encryption scheme to be probabilistic, which the new scheme is not.

A problem that needs to be addressed further is the choice of the private basis R. There is no specific reason to choose the basis in this way, one could possibly choose the private basis randomly and reduce it using LLL or some other reduction algorithm. This turns out to be good way to obtain a good basis. If we would try using LLL on the public basis HNF(R) it would take much longer and the correction radius would still be much smaller than the one of R. Micciancio shows by experimental results that when the dimension gets larger running LLL on the random matrix results in a correction radius of size n/2 but if we run LLL on the Hermite normal form the correction radius tends to zero.

So this new trapdoor function is at least as secure as the GGH trapdoor function. For the same level of security, both the size efficiency and the time efficiency is drastically improved. Thus, we can make the scheme more secure without making it impractical.

The choice of deterministic procedures instead of randomized procedures makes the scheme easier to implement and analyze.

In order to be able to compete with RSA, the key sizes need to be even smaller than those achieved through this scheme. The public key size cannot be further reduced if we do not consider lattices of special structures.

6 Computational Experiments

We want to see if there is a difference between decrypting GGH using the method of Nguyen or trying to decrypt it without that method. Nguyen shows theoretically that the GGH decryption problem can be simplified to a special case of the closest vector problem where the error vector has entries in $\{\pm\frac{1}{2}\}^n$. We are interested to know whether solving this special case is faster or not. Using Mathematica for encryption and decryption using the embedding technique (see section 2.4) for lattices of small dimensions (the running time for larger dimensions is too long) we want to see if it is generally faster. In table 1 below you can see the result.

In these dimensions one cannot see an advantage when solving this special case of the CVP. However there might be an advantage when solving the CVP in higher dimensions. The techniques to break the system are more likely to work with a smaller error vector. Also in the higher dimensions,

n	Solving GGH, $\sigma = 3$	Solving GGH, error vector with entries in $\{\pm\frac{1}{2}\}^n$
10	0.984375	1.01563
15	3.40625	3.64063
20	11.0156	11
25	24.3280	24.4063
30	44.4375	44.2031
35	83.5625	81.8594
40	178.25	178.188
45	344.141	336.078
50	516.891	529.188

Table 1: Time in seconds to solve GGH

that Nguyen breaks, he uses both LLL and BKZ for it to be possible. He stresses that it is not necessary to perform a complete BKZ reduction, one only does so until one finds the correct solution.

Supposedly, there is an advantage of solving the shortest vector problem using the embedding technique when the gap is large. The idea is that the error vector is the smallest vector. If the smallest vector is decreased then the gap will be larger and it will be easier to find the shortest vector. And then use it to solve the closest vector problem. It is possible that this property is not an advantage in these small dimensions, the differences between the size of the different error vectors might be minimal.

Hence, in these small dimensions we cannot se any advantage of using Nguyens method. His method also entails solving the problem and finding a solution modulo 2σ before solving the special case of the closest vector problem. This would make the method slower than than classical approach for these dimensions.

7 Discussion

We have seen how the original GGH scheme is built up and how Nguyen attacked it to simplify it. This means that the GGH, the original version, cannot be used unless used with a lattice of a very high dimension. These dimensions would make it too inefficient to use.

Nguyen's attack was built on a theoretical flaw in the encryption function. Then he showed experimentally that with high probability one finds few solutions of the plaintext modulo 2σ, which means that the flaw he found

theoretically can most likely be used as an advantage in practice. And then he can use this information to solve the simplified version of the closest vector problem which, according to him, is much faster than solving the original CVP. He concludes that unless the lattice is of a high dimension, the scheme is not secure. However it might be possible to make improvements making it secure and efficient enough to use in practice.

Micciancio then improves the scheme so that it first of all leaks no information about the private key. This improvement also makes the scheme easier to implement since computing the public basis is now deterministic instead of randomized as before. It is generally hard to implement random functions which makes the deterministic approach better. Furthermore, the Hermite normal form is also the provably hardest basis to transform into a good basis [9]. He then encodes the message into the error vector. Then e is encrypted to $f(e) = e \bmod B$. And decrypting means finding the closest vector to the point e modulo B. This should not be possible using B if it is not reduced first. For this encryption there does not seem to be a special case that makes it the CVP easier to solve which means that the decryption cannot be simplified. But even though all of these improvements have been made, it is still not practical because he concludes that for the scheme to be able to compete with RSA it needs to have an even smaller key. This is not possible with the improvements that he has made. Further improvements must be made if GGH is going to be of use practically when it comes to either the efficiency of implementing the scheme or the security.

Since Micciancio published his improvements in 2001, several other propositions of improvements have been made. In 2012 Masayumi Yoshino and Noburo Kunihiro [9] made an improvement of the original GGH cryptosystem using some of the properties used by Micciancio. The use of a public basis of the Hermite normal form is inherited to this scheme from Micciancio. And the idea of this scheme is to choose the entries of the error vector differently depending on some conditions and in this way making the error vector larger and harder to attack. Their conclusion is that they manage to make the scheme more secure.

8 Conclusion

The orginal GGH encryption scheme proposed in 1997 is not a secure in the dimensions proposed and if the dimension is increased the scheme cannot be of practical use. This version of the scheme is therefore dead.

There have been improvements made since 1999. Micciancio's improvements in 2001 makes the scheme secure and the decreases the key size needed. But it is not enough to compete with cryptosystems used today such as RSA. This means that in practice it is not useful. There have been suggestions for further improvements since then, making the scheme secure. But the question is; can any be of practical use?

In the article "GGH may not be dead after all" [1], Charles F. de Barros and L. Menasché Schechter discuss that the scheme introduced in [9] may be of practical use if some of the conditions are adjusted so that it will work better when implemented.

If one keeps developing the improvements made so far and take them all into account, then there may be a way to change this scheme so that it can be used in practice.

References

[1] C. F. de Barros and L. Menasché Schechter. GGH may not be dead after all. In *XXXV Congresso Nacional de Matemática Aplicada e Computacional - CNMAC 2014*. Sociedade Brasileira de Mathemática Aplicada e Computacional - SBMAC, 2014.

[2] S. D. Galbraith. 18- Algorithms for the closest and shotest vector problems. In *Mathematics of Public Key Cryptography*, first edition. Cambridge University Press, 2012.

[3] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 112-131. Springer-Verlag, 1997.

[4] J. Hoffstein, J. Pipher, and J. H. Silverman. Lattices and Cryptography. In *An Introduction to Mathematical Cryptography*, second edition. Springer-Verlag, 2014.

[5] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficeints. In *Mathematische Annalen*, volume 261, pages 515-534. Springer-Verlag, 1982.

[6] D. Micciancio. Improving Lattice Based Cryptosystems Using the Hermite Normal Form. In *CaLC*, *Lecture Notes in Computer Science* 2146, pages 126-145. Springer-Verlag, 2001.

[7] P. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *Advances in Cryptology -CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 288-304. Springer-Verlag, 1999.

[8] C. P. Schnorr and M. Euchner.Lattice basis reduction: improved practical algorithms and solving subset sum problems. In *Mathematical Programming*, volume 66, issue1-3, pages 181-199. Springer-Verlag, 1994.

[9] M. Yoshino and N. Kunihiro. Improving GGH Cryptosystem for Large Error Vector. In *2012 International Symposium on Information Theory and its Applications*,pages 416-420. IEEE, 2012.