# Cryptosystems and LLL

Thomas Baignères
EPFL - LASEC
thomas.baigneres@epfl.ch

## 1 Introduction

Since the late 70's, several public key cryptographic algorithms have been proposed. Diffie and Hellman [4] first came with this concept in 1976. Since that time, several other public key cryptosystems were invented, such as the well known RSA [22], ElGamal [5] or Rabin [21] cryptosystems. Roughly, the scope of these algorithms is to allow the secure exchange of a secret key that will later on be used to encrypt a larger amount of data. All these algorithms share one particularity, namely that their security rely on some mathematical problem which is *supposed* to be hard (computationally speaking) to solve. For example, it is well known that the ability to factorize easily the product of two large primes without any indication about the primes, would lead to break the RSA cryptosystem. Since those early days, most of the assymetric encryption algorithms that have been proposed relied on the same hard mathematical problems. Yet, it is well accepted that we should not *put all the cryptographic eggs in one basket*. This is the reason why Goldreich, Goldwasser, and Halevi proposed in 1997 (exactly 10 years after RSA) a new public-key cryptosystem [7] which security was based on lattice reduction problems. This cryptographic schemes is known as the GGH cryptosystem. Unfortunately, only two years later, Nguyen proposed an attack against GGH [15] which proved that in practice, GGH would not provide the security it originally claimed to have. The attack involved a so called lattice reduction algorithm, known as LLL [12].

The aim of this work is to provide the necessary background to understand the GGH cryptosystem and the attack that goes with it. The basis reduction algorithm LLL has many other applications in cryptography. As an example, we will review a very recent attack [17] against the GNU Privacy Guard software [9], which is a widely used free implementation of Zimmermann's famous software [20]. The necessary background about lattices, LLL, ... will be recalled in sections 2 and 3. Section 4 will review the GGH cryptosystem and Nguyen's attack against it. Finally, Section 5 will show how lattice reduction techniques can break the ElGamal [5] digital signature scheme implementation in GPGv1.2.3.

## 2 A Survey on Lattices and Related Issues

### 2.1 Lattices

If $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$ are elements of $\mathbb{R}^n$, we denote by $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^n u_i v_i$ their scalar product. We also denote $\| \mathbf{u} \|$ the Euclidean norm of $\mathbf{u}$, i.e., $\| \mathbf{u} \| = \sqrt{\langle \mathbf{u}, \mathbf{u} \rangle} = \sqrt{\sum_i u_i^2}$. Finally, $\| \mathbf{u} \|_\infty = \max_i |u_i|$ and $\| \mathbf{u} \|_1 = \sum_{i=1}^n |u_i|$ respectively denote the infinity norm and the $L^1$-norm of $\mathbf{u}$.

**Definition 1.** Let $\mathbf{f}_1, \ldots, \mathbf{f}_d$ be linearly independent vectors of $\mathbb{R}^n$. Then

$$\mathcal{L} = \sum_{i=1}^{d} \mathbb{Z}\mathbf{f}_i = \left\{ \sum_{i=1}^{d} u_i \mathbf{f}_i \ \mid \ u_i \in \mathbb{Z} \right\}$$

is called a *lattice*. The $\mathbf{f}_i$'s are a *basis* of the lattice $\mathcal{L}$. We denote a lattice $\mathcal{L}$ generated by $\mathbf{f}_1, \ldots, \mathbf{f}_d$ by $\mathcal{L}(\mathbf{f}_1, \ldots, \mathbf{f}_d)$. If the $\mathbf{f}_i$'s are considered like the rows of a $d \times n$ matrix $\mathsf{F}$, then

$$\mathcal{L} = \left\{ \mathbf{u}\mathsf{F} \ \mid \ \mathbf{u} \in \mathbb{Z}^d \right\}.$$

In that case, we also denote the lattice $\mathcal{L}(\mathsf{F})$ and call $\mathsf{F}$ a basis for the lattice $\mathcal{L}$.

It can be shown that a lattice basis always has the same number of linearly independent vectors. This number is called the *rank* of the lattice (and is equal to $d$ in the preceding definition). In this work, we restrict ourselves to *full-ranked* lattices, that is, lattices of rank equal to $n$. In the subsequent, we therefore consider a (full-ranked) lattice $\mathcal{L}$ of basis $\mathbf{f}_1, \ldots, \mathbf{f}_n \in \mathbb{R}^n$. The determinant $\det(\mathcal{L})$ of $\mathcal{L}$ is defined by

$$\det(\mathcal{L}) = \left| \det \begin{pmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_n \end{pmatrix} \right|,$$

where the $\mathbf{f}_i$'s are considered as row vectors. To see why the determinant of a lattice is well defined, it is sufficient to note that two different basis $\mathbf{f}_1, \ldots \mathbf{f}_n$ and $\mathbf{g}_1, \ldots, \mathbf{g}_n$ of the same lattice $\mathcal{L}$ can be related by a unimodular matrix. More precisely, if we denote

$$\mathsf{F} = \begin{pmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_n \end{pmatrix} \quad \text{and} \quad \mathsf{G} = \begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_n \end{pmatrix},$$

then there exists some $n \times n$ matrix $\mathsf{P}$, such that $\det(\mathsf{P}) = \pm 1$, and such that

$$\mathsf{F} = \mathsf{P} \times \mathsf{G}\,.$$

Consequently, the determinant of a lattice $\mathcal{L}$ is independent of the choice of the basis. In two dimensions, it has a very simple interpretation, as it simply is the area of the parallelogram defined by $(\mathbf{f}_1, \mathbf{f}_2)$ (see Figure 1). In dimension $n$, the determinant of the lattice is the volume

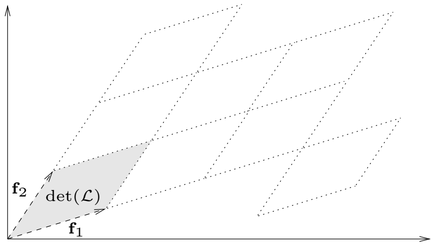

Figure 1: Interpretation of the determinant of a lattice in dimension 2

of the parallelepiped spanned by the $\mathbf{f}_i$'s. This interpretation of the determinant leads to Hadamard inequality, which states that

$$\det(\mathcal{L}) \leq \prod_{i=1}^{n} \parallel \mathbf{f_i} \parallel \ .$$

## 2.2 Dual Lattice

We consider a basis $\mathbf{f}_1, \ldots, \mathbf{f}_n \in \mathbb{R}^n$ of a lattice $\mathcal{L}$, where each $\mathbf{f}_i$'s is considered as a row vector, and we denote

$$\mathsf{F} = \begin{pmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_n \end{pmatrix} .$$

**Definition 2.** The dual lattice $\mathcal{L}^*$ of a lattice $\mathcal{L}(\mathbf{f}_1, \ldots, \mathbf{f}_n)$ is defined by

$$\mathcal{L}^* = \{\mathbf{g} \in \mathbb{R}^n \ \mid \ \langle \mathbf{g}, \mathbf{f} \rangle \in \mathbb{Z} \text{ for all } \mathbf{f} \in \mathcal{L}\} .$$

With the above notations, if $\mathbf{f}_1^*, \ldots, \mathbf{f}_n^* \in \mathbb{R}^n$ are such that

$$\langle \mathbf{f}_i^*, \mathbf{f}_j \rangle = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases}$$

then the $\mathbf{f}_i^*$'s are a basis of $\mathcal{L}^*$. The columns of the matrix $\mathsf{F}^{-1}$ are thus a basis of $\mathcal{L}^*$. Therefore, it is easy to see that $(\mathcal{L}^*)^* = \mathcal{L}$ and that $\det(\mathcal{L}) \cdot \det(\mathcal{L}^*) = 1$.

## 2.3 The Shortest Vector Problem (SVP)

When considering a lattice $\mathcal{L}$, a typical problem is to find a shortest non-zero vector in $\mathcal{L}$, i.e., a vector $\mathbf{u} \in \mathcal{L} \setminus \{\mathbf{0}\}$ such that $\parallel \mathbf{u} \parallel \leq \parallel \mathbf{v} \parallel$ for any $\mathbf{v} \in \mathcal{L} \setminus \{\mathbf{0}\}$. As the set of vector norms is discrete and lower bounded by 0, this shortest vector always exists. Its length is denoted $\parallel \mathcal{L} \parallel$. The *Shortest Vector Problem* (SVP) refers to the problem of finding a lattice vector of length $\parallel \mathcal{L} \parallel$. Finding a non-zero $\mathbf{u} \in \mathcal{L}$ such that $\parallel \mathbf{u} \parallel \leq f(n) \parallel \mathcal{L} \parallel$, where $f(n)$ is some bound depending on $n$, is the approximation problem corresponding to SVP. Very recently, it was proved [1] that SVP is NP-hard for randomized reduction. That is, a probabilistic Turing machine can solve any problem in NP in polynomial time, provided that it has access to an oracle which (when taking as an input a basis of a lattice $\mathcal{L}$) returns a solution of the shortest vector problem. Therefore, it is natural to wonder about the approximation problem of SVP. No polynomial-time algorithm is known for approximating SVP to within a polynomial factor of $n$. Nevertheless, the LLL algorithm [12] achieves to approximate SVP to within $f(n) = 2^{(n-1)/2}$ (as we shall see in Section 3) in polynomial time.

## 2.4 The Closest Vector Problem (CVP)

The *Closest Vector Problem* (CVP) is somewhat similar to SVP. In CVP, one considers a vector $\mathbf{x} \in \mathbb{R}^n$ (not necessarily in the lattice) and wants to find a point $\mathbf{u} \in \mathcal{L}$ minimizing the distance between $\mathbf{x}$ and $\mathbf{u}$, i.e., minimizing $\parallel \mathbf{x} - \mathbf{u} \parallel$. Finding a lattice vector $\mathbf{u}$ such that for all $\mathbf{v} \in \mathcal{L}$, $\parallel \mathbf{u} - \mathbf{x} \parallel \leq f(n) \parallel \mathbf{v} - \mathbf{x} \parallel$ is the approximation problem corresponding to

CVP. It has been shown [8] that SVP is not harder than CVP, that is, given an oracle which approximates CVP to within a factor $f(n)$, there exists a routine which approximates SVP to within the same factor $f(n)$ in polynomial time. Approximating CVP in polynomial time to within a factor $2^{n/2}$ can be performed via Babai's nearest plane algorithm [2]. However, in practice, the heuristic *embedding method* [7, 15] (which we shall detail in Section 3.5) is often preferred [18].

### 2.5 The Smallest Basis Problem (SBP)

We consider a lattice $\mathcal{L}$ defined by an arbitrary basis $\mathbf{f}_1, \ldots, \mathbf{f}_n$. The problem is to find the *smallest* basis for $\mathcal{L}$. Of course, the exact definition of SBP depends on the exact meaning of the word *smallest*.

**Definition 3.** Let $\mathbf{f}_1, \ldots, \mathbf{f}_n \in \mathbb{R}^n$ be the rows of the $n \times n$ matrix $\mathsf{F}$ and a basis of a lattice $\mathcal{L}$. Let $\mathbf{f}_1^*, \ldots, \mathbf{f}_n^* \in \mathbb{R}^n$ be the columns of $\mathsf{F}^{-1}$ and a basis of the dual lattice $\mathcal{L}^*$ of $\mathcal{L}$. The *size* of the basis is defined by

$$\mathsf{size}(\mathbf{f}_1, \ldots, \mathbf{f}_n) = \frac{\prod_{i=1}^n \parallel \mathbf{f}_i \parallel}{\det(\mathcal{L})}.$$

The *dual size* of $\mathsf{F}$ is the size of $\mathsf{F}^{-1}$ and is denoted

$$\mathsf{size}^*(\mathbf{f}_1, \ldots, \mathbf{f}_n) = \frac{\prod_{i=1}^n \parallel \mathbf{f}_i^* \parallel}{\det(\mathcal{L}^*)} = \det(\mathcal{L}) \prod_{i=1}^n \parallel \mathbf{f}_i^* \parallel .$$

In our case, SBP will correspond to the problem of finding the basis of smallest size. We note that, according to Hadamard inequality, the size of a basis is always greater than 1, with equality if the basis is orthogonal. This is the reason why in [7], the size of a basis is referred to as the *orthogonality defect* of the basis. It has been shown [10] that SBP is NP-hard.

## 3 Approximating SVP and CVP with LLL

### 3.1 Orthogonal Basis

We note that finding a shortest non-zero vector is trivial whenever the basis $\mathbf{f}_1, \ldots, \mathbf{f}_n$ of the lattice is orthogonal (i.e., $\langle \mathbf{f}_i, \mathbf{f}_j \rangle = 0$ for all $i \neq j$).

**Lemma 1.** *If $\mathbf{f}_1, \ldots, \mathbf{f}_n$ is an orthogonal basis of $\mathcal{L}$, the shortest vector of the basis is a shortest non-zero vector of $\mathcal{L}$.*

*Proof.* Suppose $\langle \mathbf{f}_i, \mathbf{f}_j \rangle = 0$ for all $i \neq j$ and suppose that the $\mathbf{f}_i$'s are ordered such that $\parallel \mathbf{f}_1 \parallel \leq \cdots \leq \parallel \mathbf{f}_n \parallel$. A shortest non-zero vector $\mathbf{x}$ of $\mathcal{L}$ can be written $\mathbf{x} = \sum_{i=1}^n \lambda_i \mathbf{f}_i$, where $\lambda_i \in \mathbb{Z}$. Therefore,

$$\begin{aligned} \parallel \mathbf{x} \parallel^2 &= \langle \mathbf{x}, \mathbf{x} \rangle \\ &= \sum_{i,j=1}^n \lambda_i \lambda_j \langle \mathbf{f}_i, \mathbf{f}_j \rangle \\ &= \sum_{i=1}^n \lambda_i^2 \langle \mathbf{f}_i, \mathbf{f}_i \rangle \qquad \text{(as the basis is orthogonal)} \\ &= \sum_{i=1}^n \lambda_i^2 \parallel \mathbf{f}_i \parallel^2. \end{aligned}$$

Therefore, as $\mathbf{f}_1 \in \mathcal{L}$ and as $\mathbf{x}$ is the shortest vector, we have:

$$\parallel \mathbf{x} \parallel^2 \leq \parallel \mathbf{f}_1 \parallel^2 \quad \Rightarrow \quad (\lambda_1^2 - 1) \parallel \mathbf{f}_1 \parallel^2 + \sum_{i=2}^{n} \lambda_i^2 \parallel \mathbf{f}_i \parallel^2 \leq \ 0,$$

so that, as $x \neq 0$, $(\lambda_1^2, \lambda_2^2, \ldots, \lambda_n^2) = (1, 0, \ldots, 0)$, and thus $(\lambda_1, \lambda_2, \ldots, \lambda_n) = (\pm 1, 0, \ldots, 0)$. As this implies that $\mathbf{x} = \pm \mathbf{f}_1$, this concludes the proof. □

The preceding lemma obviously solves the shortest vector problem whenever the basis of the lattice is orthogonal. A general solution would therefore be to use the Gram-Schmidt orthogonalization process in order to find an orthogonal basis for the lattice. But this is not always possible, as the orthogonal basis that one obtains after the process is not always a basis of the lattice. Nevertheless, finding an *almost* orthogonal basis for a lattice is possible.

## 3.2 The Gram-Schmidt Orthogonalization Process

Given a basis $\mathbf{f}_1, \ldots, \mathbf{f}_n$ of $\mathbb{R}^n$, the Gram-Schmidt orthogonalization process computes an orthogonal basis $\mathbf{f}_1^\star, \ldots, \mathbf{f}_n^\star$. Figure 2 illustrates the process in dimension 2. Considering an arbitrary basis $\mathbf{f}_1, \mathbf{f}_2 \in \mathbb{R}^2$, it is easy to see that an orthogonal basis is obtained by taking

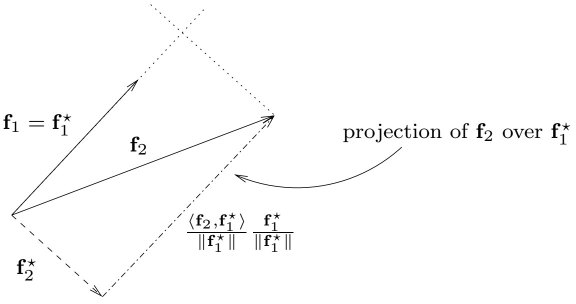
$\mathbf{f}_1 = \mathbf{f}_1^\star$
$\mathbf{f}_2$
projection of $\mathbf{f}_2$ over $\mathbf{f}_1^\star$
$\frac{\langle \mathbf{f}_2, \mathbf{f}_1^\star \rangle}{\|\mathbf{f}_1^\star\|} \frac{\mathbf{f}_1^\star}{\|\mathbf{f}_1^\star\|}$
$\mathbf{f}_2^\star$

Figure 2: Gram-Schmidt orthogonalization process in dimension 2

$$\begin{cases} \mathbf{f}_1^\star = \mathbf{f}_1 \\ \mathbf{f}_2^\star = \mathbf{f}_2 - \frac{\langle \mathbf{f}_2, \mathbf{f}_1^\star \rangle}{\|\mathbf{f}_1^\star\|^2} \mathbf{f}_1^\star. \end{cases}$$

The process in dimension $n$ is similar, and described by Algorithm 1. In matrix representation,

**Algorithm 1** The Gram-Schmidt orthogonalization process

```
Require: an arbitrary basis $\mathbf{f}_1, \ldots, \mathbf{f}_n \in \mathbb{R}^n$.
Ensure: an orthogonal basis $\mathbf{f}_1^\star, \ldots, \mathbf{f}_n^\star$ of the vector space spanned by $\mathbf{f}_1, \ldots, \mathbf{f}_n$.
$\mathbf{f}_1^\star = \mathbf{f}_1$
for $i = 2$ to $n$ do
    $\mathbf{f}_i^\star = \mathbf{f}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{f}_j^\star$ where $\mu_{i,j} = \frac{\langle \mathbf{f}_i, \mathbf{f}_j^\star \rangle}{\|\mathbf{f}_j^\star\|^2}$
end for
```

we obtain
$$\begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \vdots \\ \mathbf{f}_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots\dots\dots & & & 0 \\ \mu_{2,1} & 1 & 0 & \dots\dots & & 0 \\ \mu_{3,1} & \mu_{3,2} & 1 & 0 & \dots & 0 \\ \vdots & & & & & \vdots \\ \mu_{n,1} & \dots\dots\dots\dots\dots & & & & 1 \end{pmatrix} \times \begin{pmatrix} \mathbf{f}_1^\star \\ \mathbf{f}_2^\star \\ \mathbf{f}_3^\star \\ \vdots \\ \mathbf{f}_n^\star \end{pmatrix},$$

which we denote $\mathsf{F} = \mathsf{M} \times \mathsf{F}^\star$, and where $\mathsf{M}$ is called the Gramian matrix. As the elements $\mu_{i,j}$ of the matrix $\mathsf{M}$ are not necessarily in $\mathbb{Z}$, the $\mathbf{f}_i^\star$'s may not be lattice vectors.

### 3.3 Reduced Basis

**Theorem 1.** *With the above notations, let* $\mathcal{U}_k = \sum_{i \le k} \mathbb{R}\mathbf{f}_i$, *i.e.,* $\mathcal{U}_k$ *is the vector space spanned by* $\mathbf{f}_1, \dots, \mathbf{f}_k$. *Then,*

1. $\mathcal{U}_k = \sum_{i \le k} \mathbb{R}\mathbf{f}_i^\star$
2. $\mathbf{f}_k^\star$ *is the projection of* $\mathbf{f}_k$ *onto* $\mathcal{U}_{k-1}^\perp$, *so that* $\| \mathbf{f}_k^\star \| \le \| \mathbf{f}_k \|$
3. $\det(\mathsf{F}) = \det(\mathsf{F}^\star)$.

Note that the second result of the preceding theorem comes from the fact that $\mathbf{f}_k^\star$ is a *projection* of $\mathbf{f}_k$, so that its length can only decrease during that process. The third point is a direct implication of the matrix representation of the basis vectors.

**Theorem 2.** ***(Hadamard inequality)*** *Let* $\mathbf{f}_1, \dots, \mathbf{f}_n \in \mathbb{R}^n$ *be a basis a lattice* $\mathcal{L}$. *Then,*

$$\det(\mathcal{L}) \le \| \mathbf{f}_1 \| \cdots \| \mathbf{f}_n \| \le n^{n/2} \max_{i=1,\dots,n} \| \mathbf{f}_i \|_\infty .$$

*Proof.* We have

$$\begin{aligned} \det(\mathcal{L}) &= |\det(\mathsf{F})| && \text{(by definition)} \\ &= |\det(\mathsf{F}^\star)| && \text{(by Theorem 1)} \\ &= \| \mathbf{f}_1^\star \| \cdots \| \mathbf{f}_n^\star \| && \text{(as the } \mathbf{f}_i^\star\text{'s are orthogonal)} \\ &\le \| \mathbf{f}_1 \| \cdots \| \mathbf{f}_n \| && \text{(as } \| \mathbf{f}_k^\star \| \le \| \mathbf{f}_k \| \text{ by Theorem 1).} \end{aligned}$$

The second inequality simply comes from the fact that $\| \mathbf{f}_k \| \le n^{1/2} \| \mathbf{f}_k \|_\infty$. □

It is now possible to give a lower bound on the length of a shortest vector of a lattice, by mean of the basis obtained after the Gram-Schmidt orthogonalization process.

**Proposition 1.** *Let* $\mathbf{f}_1, \dots, \mathbf{f}_n \in \mathbb{R}^n$ *be a basis of the lattice* $\mathcal{L}$. *Then, for all nonzero* $\mathbf{f} \in \mathcal{L}$,

$$\min_{i=1,\dots,n} \| \mathbf{f}_i^\star \| \le \| \mathbf{f} \|,$$

*where* $\mathbf{f}_1^\star, \dots, \mathbf{f}_n^\star$ *are the orthogonal vectors obtained using the Gram-Schmidt orthogonalization process.*

*Proof.* As $\mathbf{f} \in \mathcal{L}$, we can write $\mathbf{f} = \sum_{i=1}^{k} \lambda_i \mathbf{f}_i$, where $\lambda_i \in \mathbb{Z}$ and $k \leq n$ such that $\lambda_k \neq 0$. Therefore,

$$\begin{aligned} \mathbf{f} &= \sum_{i=1}^{k} \lambda_i \sum_{j=1}^{i} \mu_{i,j} \mathbf{f}_j^\star \qquad (\text{with } \mu_{j,j} = 1) \\ &= \lambda_k \mathbf{f}_k^\star + \sum_{1 \leq i < k} \nu_i \mathbf{f}_i^\star \quad (\text{where } \nu_i \in \mathbb{R}). \end{aligned}$$

Thus, as the $\mathbf{f}_i^\star$'s are orthogonal,

$$\begin{aligned} \| \mathbf{f} \|^2 &= \lambda_k^2 \| \mathbf{f}_k^\star \|^2 + \textstyle\sum_{1 \leq i < k} \nu_i^2 \| \mathbf{f}_i^\star \|^2 \\ &\geq \| \mathbf{f}_k^\star \|^2 \qquad\qquad\qquad\qquad (\text{as } \lambda_k \in \mathbb{Z} \setminus \{0\}) \\ &\geq \min_{i=1,\ldots,n} \| \mathbf{f}_i^\star \|. \end{aligned}$$

□

If the $\mathbf{f}_i^\star$'s always were lattice vectors, Proposition 1 would solve SVP, as the smallest vector of the orthogonal basis would be a shortest vector of the lattice. But it is not the case.

**Definition 4.** Let $\mathbf{f}_1, \ldots, \mathbf{f}_n$ be linearly independent vectors of $\mathbb{R}^n$. Let $\mathbf{f}_1^\star, \ldots, \mathbf{f}_n^\star$ be the corresponding Gram-Schmidt orthogonalized basis. The basis $\mathbf{f}_1, \ldots, \mathbf{f}_n$ is said to be *reduced* if

$$\| \mathbf{f}_i^\star \|^2 \leq 2 \| \mathbf{f}_{i+1}^\star \|^2$$

for $i = 1, \ldots, n-1$.

Using the notion of *reduced* basis, it is possible to give a bound on a shortest vector. Although it is weaker than the one given by Proposition 1, this bound has the advantage of being function of lattice vectors.

**Theorem 3.** *If* $\mathbf{f}_1, \ldots, \mathbf{f}_n$ *is a reduced basis of a lattice* $\mathcal{L}$*, then, for all nonzero vector* $\mathbf{f} \in \mathcal{L}$*,*

$$\| \mathbf{f}_1 \| \leq \ 2^{\frac{n-1}{2}} \| \mathbf{f} \| \ .$$

*Proof.* As $\mathbf{f}_1, \ldots, \mathbf{f}_n$ is a reduced basis, we have

$$\| \mathbf{f}_1 \| = \| \mathbf{f}_1^\star \| \leq 2^{\frac{1}{2}} \| \mathbf{f}_2^\star \| \leq 2 \| \mathbf{f}_3 \| \leq \cdots \leq 2^{\frac{i-1}{2}} \| \mathbf{f}_i^\star \| \leq \cdots \leq 2^{\frac{n-1}{2}} \| \mathbf{f}_n^\star \| \ .$$

Thus, for all $i = 1, \ldots, n$, we have

$$\| \mathbf{f}_1 \| \leq 2^{\frac{i-1}{2}} \| \mathbf{f}_i^\star \| \leq 2^{\frac{n-1}{2}} \| \mathbf{f}_i^\star \|,$$

so that

$$\| \mathbf{f}_1 \| \leq 2^{\frac{n-1}{2}} \min_{i=1,\ldots,n} \| \mathbf{f}_i^\star \| \ .$$

Proposition 1 allows to conclude. □

We conclude that if, given an arbitrary basis of a lattice $\mathcal{L}$, it is possible compute a reduced basis of this lattice in a reasonable amount of time, then SVP is approximated to within a factor $2^{\frac{n-1}{2}}$ by the first vector of the reduced basis.

**Algorithm 2** The LLL basis reduction algorithm

```
Require: an arbitrary basis $\mathbf{f}_1, \ldots, \mathbf{f}_n \in \mathbb{R}^n$ of a lattice $\mathcal{L}$.
Ensure: $\mathbf{f}_1, \ldots, \mathbf{f}_n$ is a reduced basis of $\mathcal{L}$.
Compute $\mathbf{f}_1^\star, \ldots, \mathbf{f}_n^\star$, the Gram-Schmidt orthogonal basis of $\mathbf{f}_1, \ldots, \mathbf{f}_n$.
$i \leftarrow 2$
while $i \leq n$ do
  for $j = i-1, \ldots, 1$ do
    $\mathbf{f}_i \leftarrow \mathbf{f}_i - \lceil \mu_{ij} \rfloor$
    Update the $\mathbf{f}_i^\star$'s
    if $i > 1$ and $\parallel \mathbf{f}_{i-1}^\star \parallel > 2 \parallel \mathbf{f}_i^\star \parallel$ then
      Swap $\mathbf{f}_{i-1}$ and $\mathbf{f}_i$ and update the $\mathbf{f}_i^\star$'s
      $i \leftarrow i-1$
    else
      $i \leftarrow i+1$
    end if
  end for
end while
```

### 3.4 Basis Reduction with LLL

The LLL algorithm [12] solves the problem of computing a reduced basis of a lattice given an arbitrary basis. It is described by Algorithm 2, where $\lceil \mu_{ij} \rfloor$ denotes the integer closest to $\mu_{ij}$, i.e., $\lceil \mu_{ij} \rfloor = \lfloor \mu_{ij} + \frac{1}{2} \rfloor$.

**Theorem 4.** *Given an arbitrary basis of a lattice $\mathcal{L}$, the* LLL *algorithm computes a reduced basis of $\mathcal{L}$ in polynomial time.*

*Proof.* For a proof of this result, see [25]. □

We conclude that SVP is approximated to within a factor $2^{\frac{n-1}{2}}$ by the first vector of a reduced basis, which can be computed in polynomial time with LLL. The next section shows how LLL may sometimes be used to approximate CVP.

### 3.5 The Embedding Method

The embedding method [7, 15] is an heuristic technique used to approximate CVP by reducing it to SVP. Let $\mathcal{L}$ be a lattice of basis $\mathbf{f}_1, \ldots, \mathbf{f}_n$, also denoting the rows of the $n \times n$ matrix $\mathsf{F}$, and let $\mathbf{x} \in \mathbb{R}^n$ (not necessarily in the lattice). The embedding method constructs a lattice $\mathcal{L}'$ of dimension $n+1$ defined by the rows of the matrix

$$\mathsf{F}' = \left( \begin{array}{c|c} \mathsf{F} & \mathbf{0} \\ \hline \mathbf{x} & 1 \end{array} \right).$$

Considering that $\mathcal{L}$ and $\mathcal{L}'$ almost have the same dimension, and as $\det(\mathcal{L}') = \det(F') = \det(F) = \det(\mathcal{L})$, we hope that $\parallel \mathcal{L} \parallel \approx \parallel \mathcal{L}' \parallel$. Let $\mathbf{u} \in \mathcal{L}$ be a point minimizing the distance to $\mathbf{x}$, and let $\mathbf{u} = \sum_i u_i \mathbf{f}_i$. Then

$$(-u_1, \cdots, -u_n, 1) \times \left( \begin{array}{c|c} \mathsf{F} & \mathbf{0} \\ \hline \mathbf{x} & 1 \end{array} \right) = (\mathbf{x} - \mathbf{u}, 1)$$

is a short vector of $\mathcal{L}'$, which we actually hope to be a *shortest* vector. If it is, approximating SVP in $\mathcal{L}'$ leads to an approximation of CVP in $\mathcal{L}$.

# 4 Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem

The Goldreich-Goldwasser-Halevi (GGH) Cryptosystem [7] was introduced at the CRYPTO'97 conference. It was based on the computational intractability of typical lattice problems and was considered as an alternative to other classical public key cryptosystems, which security often relies on the intractability of integer factorization or discrete log computation in finite integer rings. Only two years after its publication, Nguyen showed [15] that the GGH cryptosystem does not ensure the security it initially seemed to provide. In the following sections, we will briefly discuss the GGH trapdoor function (on which the GGH encryption scheme is based), and review Nguyen's attack against it.

## 4.1 The GGH Trapdoor Function

We consider two basis $\mathsf{B}$ and $\mathsf{R}$ of the same lattice $\mathcal{L}$. The basis $\mathsf{R}$ is chosen such that its dual size is small (we will denote $\mathsf{size}^*(\mathsf{R})$ the dual size of a basis $\mathsf{R}$) and is kept secret. The basis $\mathsf{B}$ is derived from $\mathsf{R}$ in way such that $\mathsf{size}^*(\mathsf{B})$ is large, and is made public. The public basis $\mathsf{B}$ together with a positive real number $\sigma$ define the function

$$\begin{array}{cccc} f_{\mathsf{B},\sigma}: & \mathbb{R}^n \times \{\pm\sigma\}^n & \longrightarrow & \mathbb{R}^n \\ & (\mathbf{u},\mathbf{e}) & \longmapsto & \mathbf{u}\mathsf{B}+\mathbf{e}\ . \end{array}$$

We consider $\mathbf{u}$ as an element with coordinates chosen uniformly at random from the range $\{-n,\ldots,n\}$, and $\mathbf{e}$ as a uniformly distributed random element of $\{\pm\sigma\}^n$. The function $f_{\mathsf{B},\sigma}$ can then be evaluated in $(\mathbf{u},\mathbf{e})$:

$$\mathbf{c} = f_{\mathsf{B},\sigma}(\mathbf{u},\mathbf{e}).$$

## 4.2 Inverting the GGH Trapdoor function

When $\mathbf{c}$ is given, inverting the GGH trapdoor function corresponds to solve a CVP instance, where $\mathbf{u}$ is the lattice point (close to $\mathbf{c}$) that one wants to recover and where $\| \mathbf{e} \|$ is the distance between this lattice point and $\mathbf{c}$. Inverting $f_{\mathsf{B},\sigma}$ can be easily performed using the secret basis $\mathsf{R}$ and a technique due to Babai, called the Round-off algorithm [2]. The idea is to represent $\mathbf{c}$ in the basis $\mathsf{R}$ and then to round the coefficients (which are real values as $\mathbf{c}$ is not a lattice point) to the nearest integers. This will produce a lattice point close to $\mathbf{c}$ which should correspond to $\mathbf{u}\mathsf{B}$. Recovering $\mathbf{e}$ can then simply be done by subtracting the preceding point to $\mathbf{c}$. More precisely, we compute $\mathbf{c}\mathsf{R}^{-1}$, and then $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor$ (i.e., we round the coefficients to the nearest integers). The last computation should reveal the coordinates of $\mathbf{u}\mathsf{B}$ in the basis $\mathsf{R}$, so that $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}$ should correspond to the point $\mathbf{u}\mathsf{B}$. We recover $\mathbf{u}$ by computing $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}\mathsf{B}^{-1}$. Finally, $\mathbf{e}$ can be recovered by subtracting $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}$ to $\mathbf{c}$. Choosing a basis $\mathsf{R}$ with a small dual size and small value for $\sigma$ is capital for the inversion to work properly. What the above process should do, is to *correct* the small error introduced by adding $\mathbf{e}$ to $\mathbf{u}\mathsf{B}$.

**Lemma 2.** *The private basis* $\mathsf{R}$ *succeeds in inverting* $f_{\mathsf{B},\sigma}$ *if and only if* $\lceil \mathbf{e}\mathsf{R}^{-1} \rfloor = \mathbf{0}$.

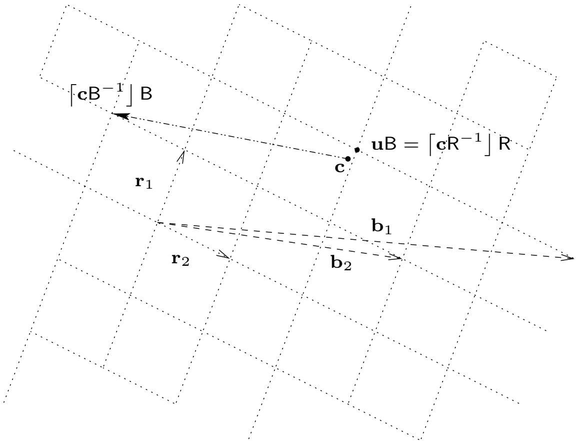
$\lceil \mathbf{c}\mathsf{B}^{-1} \rfloor \mathsf{B}$
$\mathbf{u}\mathsf{B} = \lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}$
$\mathbf{c}$
$\mathbf{r}_1$
$\mathbf{b}_1$
$\mathbf{r}_2$
$\mathbf{b}_2$

Figure 3: Comparison of B and R for inverting $f_{\mathsf{B},\sigma}$

*Proof.* Obviously, once $\mathbf{u}\mathsf{B}$ is recovered, computing $\mathbf{e} = \mathbf{c} - \mathbf{u}\mathsf{B}$ is easy. Thus, the inversion process works if and only if $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}$ corresponds to the original lattice point $\mathbf{u}\mathsf{B}$. But

$$\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R} = \lceil (\mathbf{u}\mathsf{B} + \mathbf{e})\mathsf{R}^{-1} \rfloor \mathsf{R} = \lceil \mathbf{u}\mathsf{B}\mathsf{R}^{-1} + \mathbf{e}\mathsf{R}^{-1} \rfloor \mathsf{R} = \mathbf{u}\mathsf{B} + \lceil \mathbf{e}\mathsf{R}^{-1} \rfloor \mathsf{R}$$

as $\mathbf{u}\mathsf{B}\mathsf{R}^{-1}$ only has integer coefficients, so that $\lceil \mathbf{c}\mathsf{R}^{-1} \rfloor \mathsf{R}$ is equal to $\mathbf{u}\mathsf{B}$ if and only if $\lceil \mathbf{e}\mathsf{R}^{-1} \rfloor = \mathbf{0}$. □

It seems natural to wonder if B can used instead of R in order to invert $f_{\mathsf{B},\sigma}$. The following example illustrates what happens in such a case.

*Example* 1. Take the following example in dimension 2 (illustrated on Figure 3):

$$\mathsf{R} = \begin{pmatrix} 3 & 8 \\ 8 & -4 \end{pmatrix} = \begin{pmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \end{pmatrix} \quad \text{and} \quad \mathsf{B} = \begin{pmatrix} 46 & -4 \\ 27 & -4 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}.$$

As $\det(\mathsf{R}\mathsf{B}^{-1}) = 1$, R and B are basis of the same lattice. We have $\mathsf{size}^*(\mathsf{R}) \approx 1.006$ whether $\mathsf{size}^*(\mathsf{B}) \approx 3.97$. We pick $\sigma = 1$, $\mathbf{u} = (2, 2)$ and $\mathbf{e} = (-\sigma, -\sigma) = (-1, -1)$. The evaluation of $f_{\mathsf{B},\sigma}$ in $(\mathbf{u}, \mathbf{e})$ returns $\mathbf{c} = (21, 7)$. If we try to correct $\mathbf{e}$ using R we obtain $\mathbf{e}\mathsf{R}^{-1} = (-3/19, -5/76)$, so that $\lceil \mathbf{e}\mathsf{R}^{-1} \rfloor = \mathbf{0}$. The same computation using the public basis leads to $\mathbf{e}\mathsf{B}^{-1} = (-31/76, -25/38)$, so that $\lceil \mathbf{e}\mathsf{B}^{-1} \rfloor = (0, -1)$. Figure 3 illustrates the result of the inversion process with either B or R.

It is now clear that error can be corrected by R, provided that $\sigma$ is small enough. The following theorem approximates its size.

**Theorem 5.** *Let* R *denote the private basis used to invert $f_{\mathsf{B},\sigma}$, and let $\rho$ be the maximum value of the $L^1$-norm of the columns of* $\mathsf{R}^{-1}$. *If $\sigma < \frac{1}{2\rho}$, then the private basis* R *succeeds in inverting $f_{\mathsf{B},\sigma}$.*

*Proof.* According to Lemma 2, the inversion succeeds when $\lceil \mathbf{e}\mathsf{R}^{-1} \rfloor = \mathbf{0}$. This is obviously the case when $\max_i |\langle \mathbf{e}, \mathbf{r}_i^* \rangle| < 1/2$, where $\mathbf{r}_i^*$ denotes the $i$th column of $\mathsf{R}^{-1}$. As $|\langle \mathbf{e}, \mathbf{r}_i^* \rangle| \leq \sigma \parallel \mathbf{r}_i^* \parallel_1$, we can easily obtain a sufficient condition for the inversion of the trapdoor function to be successful. □

From the preceding theorem, it is now clear that $\sigma$ shall not be chosen to big, unless $\mathsf{R}$ will not be able to recover the original lattice point, but should not be too small in order to avoid the public basis $\mathsf{B}$ to be also able to correct errors. The following theorem studies what happens when security is privileged compared to the probability of inversion errors.

**Theorem 6.** *Let* $\mathsf{R}$ *denote the private basis used to invert* $f_{\mathsf{B},\sigma}$*, and let* $\gamma/\sqrt{n}$ *be the maximum value of the infinity norm of the columns of* $\mathsf{R}^{-1}$*. Then the probability* $\mathrm{P_e}$ *that an inversion error occurs is bounded by*

$$\mathrm{P_e} \;\leq\; 2n\exp\left(-\frac{1}{8\sigma^2\gamma^2}\right)\;.$$

*Proof.* Let $\mathbf{d} = \mathbf{e}\mathsf{R}^{-1}$. According to Lemma 2, an inversion error occurs as soon as $|d_i| > \frac{1}{2}$ for some $i$. As the $d_i$'s are independent, $\mathrm{P_e} \leq n\Pr[|d_i| > \frac{1}{2}]$. Denoting $\rho_{ij}$ the $i,j$'th element of $\mathsf{R}^{-1}$, we have

$$\mathrm{E}[d_i] = \mathrm{E}\left[\sum_{j=1}^{n} e_j\rho_{ji}\right] = \sum_{j=1}^{n}\mathrm{E}[e_j\rho_{ji}] = \sum_{j=1}^{n}\mathrm{E}[e_j]\mathrm{E}[\rho_{ji}],$$

as $\mathbf{e}$ and $\mathsf{R}^{-1}$ are independent. But as each $e_j$ is uniformly distributed in $\{\pm\sigma\}$, $\mathrm{E}[e_j] = 0$, so that $\mathrm{E}[d_i] = 0$. Moreover, as the infinity norm of each column of $\mathsf{R}^{-1}$ is bounded by $\gamma/\sqrt{n}$, then $e_j\rho_{ij}$ is a random variable in $\left[-\frac{\sigma\gamma}{\sqrt{n}}, \frac{\sigma\gamma}{\sqrt{n}}\right]$. Using Hoeffding's bound (see Appendix A), we obtain

$$\Pr\left[|d_i| > \frac{1}{2}\right] < 2\exp\left(-\frac{1}{8\sigma^2\gamma^2}\right),$$

which concludes the proof. □

To give an idea of the possible parameters size, the authors considered the case where the probability of error is $10^{-5}$, the dimension $n = 120$, and for which the maximum Euclidean norm of the columns of $\mathsf{R}^{-1}$ is $1/30$. For these values, the preceding theorem gives approximatively 2.5 as an upperbound on $\sigma$. Practical values of $\sigma$ in higher dimensions could thus be around 3.

### 4.3 The Original Security Analysis

It is legitimate to wonder how hard is the GGH trapdoor function inversion problem. In the original publication, several attacks were considered. The first thing an attacker could do, is to reduce the size of the public basis $\mathsf{B}$, so that $\mathsf{B}$ will always be considered as a basis reduced via a reduction algorithm (such as LLL). The most obvious attack is to use the public basis $\mathsf{B}$ instead of the private basis $\mathsf{R}$ in the inversion of $f_{\mathsf{B},\sigma}$. This is the situation that was illustrated by Example 1. A more detailed study of this attack, together with practical experiments, lead the authors to consider it inefficient for a dimension $n$ greater than 100. A refinement of the preceding attack is to use Babai's nearest plane algorithm [2] which (just as the round-off algorithm) also approximates CVP. The nearest plane algorithm gives a much better approximation, yet attack on the GGH trapdoor with this algorithm are not feasible from dimensions around 150. Finally, the embedding method (which we detailed in Section 3.5) is another method to approximate CVP and is effective until dimensions around 120. The authors conjectured that, if the reduction algorithm used is LLL, their scheme was secure for dimensions above 150. Yet, *better* reduction algorithm do exists [23], so that their conclusion is that the GGH trapdoor function should be secure for dimensions around 250-300.

## 4.4 Introduction of Nguyen's Attack against the GGH Trapdoor Function

We suppose that the GGH trapdoor function $f_{\mathsf{B},\sigma}$ is evaluated in $(\mathbf{u}, \mathbf{e}) \in \mathbb{Z}^n \times \{\pm\sigma\}^n$ in order to obtain

$$\mathbf{c} = \mathbf{u}\mathsf{B} + \mathbf{e} \ . \tag{1}$$

Let $\mathbf{s} = (\sigma, \ldots, \sigma) \in \mathbb{Z}^n$. As $\mathbf{e} \in \{\pm\sigma\}^n$, it is clear that

$$\mathbf{c} + \mathbf{s} \equiv \mathbf{u}\mathsf{B} \pmod{2\sigma}.$$

If we let $\mathbf{v} = \mathbf{c} + \mathbf{s}$ we are left with a modular system of equations

$$\mathbf{v} \equiv \mathbf{u}\mathsf{B} \pmod{2\sigma}, \tag{2}$$

with unknown $\mathbf{u}$. Suppose this system has a unique solution, i.e., suppose that $\mathsf{B}$ is invertible modulo $2\sigma$. Let $\mathbf{u}_{2\sigma} = \mathbf{v}\mathsf{B}^{-1} \bmod 2\sigma$ denote the solution of (2). As $\mathbf{c} = \mathbf{u}\mathsf{B} + \mathbf{e}$, we have

$$\mathbf{c} - \mathbf{u}_{2\sigma}\mathsf{B} = (\mathbf{u} - \mathbf{u}_{2\sigma})\mathsf{B} + \mathbf{e}.$$

But as $\mathbf{u}_{2\sigma} = \mathbf{u} \bmod 2\sigma$, we have $2\sigma \mid \mathbf{u} - \mathbf{u}_{2\sigma}$. Therefore, we can write

$$\mathbf{u} - \mathbf{u}_{2\sigma} = 2\sigma\mathbf{u}',$$

which gives

$$\frac{\mathbf{c} - \mathbf{u}_{2\sigma}\mathsf{B}}{2\sigma} = \mathbf{u}'\mathsf{B} + \frac{\mathbf{e}}{2\sigma} \ . \tag{3}$$

We notice the similarity between (1) and (3). In (1), $\mathbf{c}$ and $\mathsf{B}$ are known, the attacker being left with a CVP instance which is hard to solve, as $\mathbf{e}$ (or more exactly $\sigma$) was chosen for this instance to be so. On the other hand, the rational point $\frac{\mathbf{c}-\mathbf{u}_{2\sigma}\mathsf{B}}{2\sigma}$ and the basis $\mathsf{B}$ are also known in (3). Yet the CVP instance the attacker has to solve is much easier, as the distance between the lattice point and the rational point is much smaller. Indeed, as $\frac{\mathbf{e}}{2\sigma} \in \{\pm\frac{1}{2}\}^n$, we have $\| \frac{\mathbf{e}}{2\sigma} \| = \frac{\sqrt{n}}{2}$, which is much smaller than $\| \mathbf{e} \| = \sigma\sqrt{n}$. If the new CVP instance is easy to solve (using for example the embedding method), the attacker will recover $\frac{\mathbf{e}}{2\sigma}$ and thus $\mathbf{e}$, so that the original CVP instance (1) is also solved.

We have seen that, provided that (2) has a unique solution which can be found, the GGH trapdoor can be inverted by an attacker. The following section studies the probability for this to happen.

## 4.5 Invertible Matrices

We consider the modular system

$$\mathbf{v} = \mathbf{u}\mathsf{B} \pmod{N}, \tag{4}$$

similar to (2), and wonder how often this system only has one single solution when $N$ is small. We will consider that the entries of $\mathsf{B} \bmod N$ are uniformly distributed in $\mathbb{Z}_N$. Clearly (4) only has one single solution if and only if $\mathsf{B}$ is invertible modulo $N$, i.e., if and only if $\det(\mathsf{B})$ is coprime with $N$. The following theorem evaluates how often this can happen.

Table 1: Ratio of invertible matrices of $\mathcal{M}_n(\mathbb{Z}_N)$, for $n = 200$.

| $N$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| ratio (%) | 28.9 | 56.0 | 28.9 | 76.0 | 16.2 | 83.7 | 28.9 | 56.0 | 22.0 |

**Theorem 7.** *Let $N$ be a positive integer, and let $p_1, \ldots, p_\ell$ be the distinct prime factors of $N$. We consider the ring of $n \times n$ invertible matrices of $\mathbb{Z}_N$. The ratio of invertible matrices is equal to*

$$\prod_{i=1}^{\ell} \prod_{k=1}^{n} (1 - p_i^{-k}) \, .$$

*Proof.* Let $\mathcal{M}_n(\mathbb{Z}_N)$ denote the ring of $n \times n$ matrices with coefficients in $\mathbb{Z}_N$. Consider the case where $N = p^\alpha$, where $p$ is a prime number and $\alpha$ is a positive integer. A matrix of $\mathcal{M}_n(\mathbb{Z}_{p^\alpha})$ is invertible modulo $p^\alpha$ if and only if its determinant is coprime with $p^\alpha$, i.e., if and only if its determinant is not divisible by $p$. Therefore, the ratio of invertible matrices in $\mathcal{M}_n(\mathbb{Z}_{p^\alpha})$ is the same as in $\mathcal{M}_n(\mathbb{Z}_p)$. The number of invertible matrices in $\mathcal{M}_n(\mathbb{Z}_p)$ is equal to the number of linearly independent families of $n$ vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$. Clearly, once $\mathbf{b}_1, \ldots, \mathbf{b}_k$ are fixed, $\mathbf{b}_{k+1}$ can be chosen among $\mathcal{M}_n(\mathbb{Z}_p) \setminus \{\lambda_1 \mathbf{b}_1 + \cdots + \lambda_k \mathbf{b}_k \mid \lambda_1, \ldots, \lambda_k \in \mathbb{Z}_p\}$, which is a set of cardinality $p^n - p^k$. Therefore, the number of linearly independent families of $n$ vectors is

$$\prod_{k=0}^{n-1} (p^n - p^k) = p^{n^2} \prod_{k=1}^{n} (1 - p^{-k}) \, .$$

When $N = p_1^{\alpha_1} \cdots p_\ell^{\alpha_\ell}$ involves $\ell$ distinct prime powers, the Chinese Reminder Theorem finally proves that the number of invertible matrices of $\mathcal{M}_n(\mathbb{Z}_N)$ is the product of the numbers of invertible matrices of each $\mathcal{M}_n(\mathbb{Z}_{p_i^{\alpha_i}})$. This leads to the announced result. □

For the considered dimensions of $n$, the ratio of invertible matrices is almost constant. As an illustration, Table 1 gives examples of ratios for different typical values of $N$. Recall that in our case, practical values of $N = 2\sigma$ are around 6.

### 4.6 Further Notes and Conclusion

We conclude from the preceding study, that GGH is not as secure as it first seemed to be. The original hard to solve CVP instance can be reduced to an easier CVP instance which will (most of the time) lead to an information that can help to solve the original CVP instance. In [15, 16], the discussion is not limited to the case where (2) only has one solution. Another study, similar to the one presented in Section 4.5, evaluates the size of the kernel of $n \times n$ matrices in $\mathbb{Z}_p$. For the particular case where $N = 2\sigma = 6$, it lead to the results shown on Table 2. We see that, even if the matrix $\mathsf{B}$ is not invertible, it has great chances of having a small kernel. If we go back to (2), we see that two distinct solutions $\mathbf{u}$ and $\mathbf{u}'$ are such that

$$(\mathbf{u} - \mathbf{u}')\mathsf{B} \equiv 0 \pmod{2\sigma} \, ,$$

so that the solutions of (2) can derived from the kernel of $\mathsf{B}$ and one particular solution of the system (note that computing the kernel of a matrix modulo a prime can be done in polynomial

Table 2: Ratio of matrices of $\mathcal{M}_n(\mathbb{Z}_6)$ of small kernel, for $n = 200$.

| Kernel Size | 1 | 2 | 3 | 4 | 6 | Other Cases |
|---|---|---|---|---|---|---|
| ratio (%) | 16.2 | 32.4 | 12.1 | 7.2 | 24.3 | 5.1 |

time, see [3, p.57-65]). We conclude that the number of solutions of (2) is small, so that even when $\mathsf{B}$ is not invertible, solving the hard CVP instance (1) can be done by solving a small number of easier CVP instances like (3).

Finally, after the publication of the GGH cryptosystem, a set of 5 challenges with dimensions going from 200 to 400 were published on the Internet [6]. Although GGH was conjectured secure for dimensions higher than 300, experimentations [15] allowed to break the first four challenges, including one in dimension 350.

# 5 Breaking GPGv1.2.3 Implementation of ElGamal Signature

The GNU Privacy Guard [9] is a free implementation of the OpenPGP [19] standard, which is based on Zimmermann well known software PGP [20]. Because PGP uses IDEA, a patented symmetric encryption algorithm, and as it is not sure if the source code will stay open, GnuPG was developed in the late 90s. GPG v1.2.3 was released in August 22nd, 2003. By the end of 2003, Nguyen discovered several flaws in this version of GPG [14, 17]. Basically, Nguyen's attack allows to recover the signer private key in less than a second on a PC, given only one signature of an arbitrary message, provided that the user chose ElGamal signature scheme at setup time. It seems that the flaw which is exploited in this attack has been also present in earlier version of GPG. Fortunately, choosing ElGamal as a digital signature scheme was not the default option proposed to the user, so that the attack only concerns those who explicitly choose ElGamal signature scheme. Nevertheless, Nguyen's attack lead the developers of GPG to make a new version of GPG available [11, 9], from which ElGamal signatures and ElGamal sign+encrypt keys have been removed. In the following we will only give the information needed to understand the attack against GPGv1.2.3 ElGamal signature. For more details, the reader should refer to [17]. In the next sections, we will recall how ElGamal signatures work, briefly explain how these signatures are implemented in GPGv1.2.3, and finally explain Nguyen's attack.

## 5.1 ElGamal Signature Scheme

The ElGamal [5] signature scheme is defined as follows. The public parameters are a prime number $p$ and a generator $g$ of the cyclic group $\mathbb{Z}_p^*$. To setup the public/private key pair, one take a random $x \in_{\mathsf{R}} \mathbb{Z}_{p-1}$ and compute $y = g^x \bmod p$. The private key is $x$, and the corresponding public key is $y$. Signing a message $M$ is done by first hashing the message via a cryptographic hash function $H$,

$$m = H(M) \in \mathbb{Z}_{p-1}.$$

Then a random $k \in_{\mathsf{R}} \mathbb{Z}_{p-1}^*$ allows to compute

$$\begin{aligned} a &= g^k \bmod p \\ b &= (m - ax)k^{-1} \bmod (p-1). \end{aligned}$$

Table 3: The Wiener table used to generate ElGamal public parameter $p$

| Bit-length of $p$ | 512 | 768 | 1024 | 1280 | 1536 | 1792 | 2048 | 2304 | 2560 | 2816 | 3072 | 3328 | 3584 | 3840 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $q_{\text{bit}}$ | 119 | 145 | 165 | 183 | 198 | 212 | 225 | 237 | 249 | 259 | 269 | 279 | 288 | 296 |

The signature is $\sigma = (a, b)$.

A given signature is valid if the following congruence holds:

$$y^a a^b \equiv g^m \pmod{p}.$$

Indeed, if the signature is valid, we have

$$ax + bk \equiv m \pmod{p-1},$$

so that

$$y^a a^b \equiv g^{ax} g^{bk} \equiv g^{ax+bk} \equiv g^m \pmod{p}.$$

In the subsequent we will abusively denote by $m \in \mathbb{Z}_{p-1}$ the message to be signed.

### 5.2 GPGv1.2.3 Implementation of ElGamal Signature Scheme

In GPGv1.2.3 the public parameter $p$ is a large prime chosen in way such that the complete factorization of $p-1$ is known. Moreover, all the prime factors of $\frac{p-1}{2}$ should have a bit-length greater that $q_{\text{bit}}$, which is function of the bit-length of $p$ itself. The correspondence is given by the Wiener table (see Table 3). We can see that we always have $4q_{\text{bit}} < \log_2 p$. Although the private exponent $x$ should be chosen at random in $\mathbb{Z}_{p-1}$, the developers chose to generate an $x$ of bit-length $3q_{\text{bit}}/2$, for efficiency reasons. This means that $x$, instead of having a bit-length of order $\log_2 p$, has a much smaller bit length. This particularity is at the origin of Nguyen's attack. When signing a message, GPGv1.2.3 should generate a random $k \in_{\mathsf{R}} \mathbb{Z}^*_{p-1}$. Instead, a random $k$ of bit-length $3q_{\text{bit}}/2$ is generated. Then, while $k$ is not coprime with $p-1$, $k$ is incremented. Obviously, $k$ will be much smaller than $p$ after this process, instead of being of comparable size.

### 5.3 An Attack against GPGv1.2.3 Implementation of ElGamal Signatures

We consider the case where an attacker has access to a valid signature $\sigma = (a, b)$ of a message $m \in \mathbb{Z}_{p-1}$. As the signature is valid, the following congruence should hold:

$$ax + bk \equiv m \pmod{p-1}. \tag{5}$$

The values of $x$ and $k$ are unknown. But, as we know that they approximatively are of bit-length $3q_{\text{bit}}/2$, we know (from Wiener table) that they are much smaller that $\sqrt{p}$. This lead to consider (5) as a closest vector problem instance in a two dimensional lattice. The following lemma gives all the details we need about the lattice we will use to solve (5).

**Lemma 3.** *Let $(\alpha, \beta) \in \mathbb{Z}^2$ and $n \in \mathbb{N}$. Let $d = \gcd(\alpha, n)$ and let $e = \gcd(\alpha, \beta, n)$. Let*

$$\mathcal{L} = \{(u, v) \in \mathbb{Z}^2 \mid \alpha u + \beta v \equiv 0 \pmod{n}\}.$$

*Then:*

1. *$\mathcal{L}$ is a two dimensional lattice of $\mathbb{Z}^2$.*
2. $\det(\mathcal{L}) = \frac{n}{e}$.
3. *There exists $u \in \mathbb{Z}$ such that $\alpha u + (\beta/e)d \equiv 0 \pmod{n}$.*
4. *The vectors $(n/d, 0)$ and $(u, d/e)$ form a basis of $\mathcal{L}$.*

*Proof.* For a proof of this result, see [17]. □

The preceding lemma is enough to break GPGv1.2.3 implementation of ElGamal signatures. Let

$$\mathcal{L} = \{(u,v) \in \mathbb{Z}^2 \mid au + bv \equiv 0 \pmod{p-1}\}.$$

According to Lemma 3, $\mathcal{L}$ is a two dimensional lattice. If we denote $d = \gcd(a, p-1)$ and $e = \gcd(a, b, p-1)$, Lemma 3 also states that there exists $u \in \mathbb{Z}$ such that $au + (b/e)d \equiv 0 \pmod{p-1}$. A basis of $\mathcal{L}$ is then given by the rows of

$$\mathsf{B} = \begin{pmatrix} \frac{p-1}{d} & 0 \\ u & \frac{d}{e} \end{pmatrix}$$

Once a basis is computed, we compute a pair $(x', k') \in \mathbb{Z}^2$ such that $ax' + bk' \equiv m \pmod{p-1}$. For this we can apply the extended Euclidean algorithm to find three integers $\lambda_1, \lambda_2, \lambda_3$ such that $a\lambda_1 + b\lambda_2 + (p-1)\lambda_3 = e$. Because of (5), we know that $e$ divides $m$, so that multiplying $\lambda_1$ and $\lambda_2$ by $\frac{m}{e}$ will result in good candidates for $x'$ and $k'$ respectively.

We now consider $\mathbf{l} = (x' - x, k' - k)$ and $\mathbf{t} = (x' - 2^{3q_{\text{bit}}/2-1}, k' - 2^{3q_{\text{bit}}/2-1})$. As

$$a(x'-x) + b(k'-k) \equiv (ax' + bk') + (ax + bk) \equiv 0 \pmod{p-1},$$

we have $\mathbf{l} \in \mathcal{L}$. However, $\mathbf{t} \notin \mathcal{L}$. Yet, as in GPGv1.2.3 the bit-lengths of both $x$ and $k$ can be approximated by $3q_{\text{bit}}/2$, we can approximate the distance between the two vectors $\mathbf{l}$ and $\mathbf{t}$ by

$$\| \mathbf{t} - \mathbf{l} \| = \sqrt{(x - 2^{3q_{\text{bit}}/2-1})^2 + (k - 2^{3q_{\text{bit}}/2-1})^2} \approx 2^{\frac{3q_{\text{bit}}-1}{2}}.$$

On the other hand, we know by Lemma 3 that $\det(\mathcal{L}) = \frac{p-1}{e} = \frac{p-1}{\gcd(a,b,p-1)}$, which should be (by construction) approximatively equal to $p$. Indeed, as $\frac{p-1}{2}$ is not smooth, we can suppose that $a$,$b$, and $\frac{p-1}{2}$ have no common factor. As we noticed in the Wiener table that the bit-length of $p$ is always greater that $4q_{\text{bit}}$, we deduce the following approximation:

$$\det(\mathcal{L})^{1/2} \approx \sqrt{p} > 2^{2q_{\text{bit}}}.$$

We now make an heuristic assumption, namely that $\mathbf{l}$ is the closest lattice vector of $\mathbf{t}$. Indeed, due to the huge difference size between $\| \mathbf{t} - \mathbf{l} \|$ and $\det(\mathcal{L})^{1/2}$, this assumption is natural (see Figure 4). Therefore, $\mathbf{l}$ can be recovered by solving a simple CVP instance, i.e., by computing the lattice point close to $\mathbf{t}$ (which is known). This can be done using, for example, Babai's Round-off algorithm [2], or the embedding method [7, 15] (see Section 3.5). Once the vector $\mathbf{l}$ is found, recovering $x$ and $k$, i.e., the private key and the private random value originally used to sign the message $m$, is a fairly easy task.

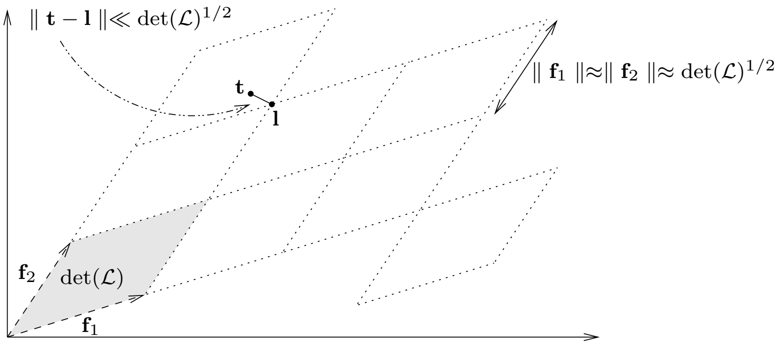
$\| \mathbf{t}-\mathbf{l} \| \ll \det(\mathcal{L})^{1/2}$
$\mathbf{t}$
$\mathbf{l}$
$\| \mathbf{f}_1 \| \approx \| \mathbf{f}_2 \| \approx \det(\mathcal{L})^{1/2}$
$\mathbf{f}_2$
$\det(\mathcal{L})$
$\mathbf{f}_1$

Figure 4: Classical heuristic assumption for CVP

### 5.4 Yet Another Attack against GPGv1.2.3 Implementation of ElGamal Signatures

Nguyen also proposed and alternative attack [17]. Let $K$ denote a large integer and let $\mathcal{L}'$ be the $4-$dimensional lattice spanned by the rows of

$$\mathbf{B}' = \begin{pmatrix} (p-1)K & 0 & 0 & 0 \\ -mK & 2^{3q_{\text{bit}}/2} & 0 & 0 \\ bK & 0 & 1 & 0 \\ aK & 0 & 0 & 1 \end{pmatrix} .$$

The vector $\mathbf{l}' = (0, 2^{3q_{\text{bit}}/2}, k, x)$ is a vector of $\mathcal{L}'$. Indeed, because of (5), we know there exists some $\lambda \in \mathbb{Z}$ such that $(p-1)\lambda - m + bk + ax = 0$. Thus

$$(\lambda, 1, k, x)\mathbf{B}' = ((p-1)\lambda K - mK + bkK + axK, 2^{3q_{\text{bit}}/2}, k, x) = (0, 2^{3q_{\text{bit}}/2}, k, x) .$$

Using the same argument as in the preceding section, we can notice that

$$\frac{\det(\mathcal{L}')^{1/4}}{\| \mathbf{l}' \|} \approx \frac{2^{-q_{\text{bit}}/8} K^{1/4}}{\sqrt{3}}$$

which is much larger than 1 provided that $K \gg 9 \cdot 2^{q_{\text{bit}}/2}$. In such a case, using similar heuristic arguments as those of the preceding section (see Figure 4), we make the assumption that $\mathbf{l}'$ is a shortest vector of $\mathcal{L}'$. This vector can usually be recovered by approximating the shortest vector of $\mathcal{L}'$ with LLL.

We implemented this attack (for $p$'s of approximatively 512 bits) using the integer LLL algorithm included in Maple [13] and concluded that, as expected, less than a second is necessary to recover the signer private key.

## 6 Conclusion

In this work we provided some illustrations of the usage of LLL in cryptography. Finally, I would like to thank Prof. Amin Shokrollahi for having proposed this work to me, and for introducing me to several great topics in algorithmic number theory.

# References

[1] M. Ajtai. The shortest vector problem in L2 is NP-hard for randomized reductions. In *Proc. 30th annual ACM symposium on Theory of computing*, pages 10–19. ACM Press, New York, NY, USA, 1998. Available at `http://www.eccc.uni-trier.de/eccc/` as TR97-047.

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

[3] H. Cohen. *A Course in Computational Algebraic Number Theory.* Springer-Verlag, 1993.

[4] W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.

[5] T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In G.R. Blakley and D. Chaum, editors, *Advances in Cryptology* - CRYPTO *'84*, volume 196, pages 10–18. Springer-Verlag, 1984.

[6] O. Goldreich, S. Goldwasser, and S. Halevi. Five challenges for braking the GGH cryptosystem. Available at `http://theory.lcs.mit.edu/~shaih/challenge.html`.

[7] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In B.S. Kaliski Jr., editor, *Advances in Cryptology* - CRYPTO *'97*, volume 1294, pages 112–131. Springer-Verlag, 1997. Available at `http://www.eccc.uni-trier.de/eccc/` as TR96-056.

[8] O. Goldreich, D. Micciancio, S. Safra, and J.-P. Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. 1999. Available at `http://www.eccc.uni-trier.de/eccc/` as TR99-002.

[9] GPG. The GNU Privacy Guard. `http://www.gnupg.org`.

[10] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, 1993.

[11] W. Koch. GnuPG's ElGamal signing keys compromised. Internet public annoucement, November 27th, 2003.

[12] A.K. Lenstra, H.W Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Ann.*, 261:513–534, 1982.

[13] Maplesoft. Maple v9.03. `http://www.maplesoft.com`.

[14] P. Nguyen. ASIACRYPT'03 rump session.

[15] P. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto'97. In M. Wiener, editor, *Advances in Cryptology* - CRYPTO *'99*, volume 1666, pages 288–304. Springer-Verlag, 1999.

[16] P. Nguyen. *La Géométrie des Nombres en Cryptologie.* PhD thesis, École Normale Supérieure, 45 rue d'Ulm, Paris 5e, Nov. 1999. Available at `http://www.di.ens.fr/~pnguyen`.

[17] P. Nguyen. Can We Trust Cryptographic Software? Cryptographic Flaws in GNU Privacy Guard v1.2.3. In C. Cachin, editor, *Advances in Cryptology* - EUROCRYPT *'04*, volume 3027, pages 555–570. Springer-Verlag, 2004.

[18] P. Nguyen and J. Stern. The two faces of lattices in cryptology. In J.H. Silverman, editor, *Cryptography and Lattices : International Conference, CaLC 2001*, volume 2146, pages 146–180. Springer-Verlag, 2001.

[19] OpenPGP. `http://www.openpgp.org`.

[20] PGP. Pretty Good Privacy. `http://www.pgp.com`.

[21] M.O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. MIT/LCS/TR-212,MIT Laboratory for Computer Science, 1979.

[22] R.L. Rivest, A. Shamir, and L.M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[23] C.P. Schnorr and H.H. Hörner. Attacking the Chor-Rivest Cryptosystem by Improved Lattice Reduction. In L.C. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology* - EUROCRYPT *'95*, volume 921, pages 1–12. Springer-Verlag, 1995.

[24] S. Vaudenay. Communication Security: An Introduction to Cryptography, 2004. Lecture notes of the *Cryptography and Security* course, EPFL.

[25] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, 2nd edition, 2003. First published 1999.

# A Hoeffding's Bound

We consider a sequence $X_1, X_2, \ldots, X_n$ of $n$ independent bounded random variables, so that there exists two real numbers $a$ and $b$ such that, for each $i = 1, \ldots, n$,

$$a \leq X_i \leq b \, .$$

Let $S_n = \sum_{i=1}^{n} X_i$. Then the Hoeffding's bound states that

$$\Pr[|S_n - \mathrm{E}[S_n]| \geq n\epsilon] \leq 2 \exp\left(\frac{-2n\epsilon^2}{b-a}\right) \, .$$