Odd Manhattan's Algorithm Specifications and Supporting Documentation

Thomas PLANTARD

Institute of Cybersecurity and Cryptology
School of Computing and Information Technology
Faculty of Engineering and Information Sciences
University of Wollongong

1 Background

Definition 1 (Lattice). *A lattice $\mathcal{L}$ is a discrete sub-group of $\mathbb{R}^n$, or equivalently the set of all the integral combinations of $d \leq n$ linearly independent vectors over $\mathbb{R}$.*

$$\mathcal{L} = \mathbb{Z}\, b_1 + \cdots + \mathbb{Z}\, b_d, \quad b_i \in \mathbb{R}^n.$$

$B = (b_1, ..., b_d)$ is called a basis of $\mathcal{L}$ and d, the dimension of $\mathcal{L}$, noted $dim(\mathcal{L})$.
$\mathcal{L}(B)$ refers to the lattice generating by a basis B.
A lattice of dimension $d = n$ is called full-rank.
A lattice $\mathcal{L} \subseteq \mathbb{Z}^n$ is called integer lattice.

The *determinant* of a lattice $\mathcal{L}$, noted $det(\mathcal{L})$, can be computed by $det(\mathcal{L}) = \sqrt{det\,(BB^T)}$.

Property 1 (Random Lattice). *In [6], Goldstein and Mayer characterized random lattices: if p is prime, a random full-rank integer lattice of determinant p is generated by a basis*

$$\begin{pmatrix} p & 0 & 0 & \cdots & 0 & 0 \\ a_1 & 1 & 0 & \cdots & 0 & 0 \\ a_2 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ a_{d-1} & 0 & 0 & \cdots & 1 & 0 \\ a_d & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

with a_i uniform in $[0, p-1]$.

Remark 1. *We will focus on full-rank integer lattice with prime determinant to match Goldstein-Mayer characterization.*

Definition 2 (Dual Lattice). *Let $\mathcal{L}$ a lattice, its dual lattice $\mathcal{L}^*$ is a lattice such that*

$$\mathcal{L}^* = \{u \in span(\mathcal{L}), \forall v \in \mathcal{L}, \langle u, v\rangle \in \mathbb{Z}\}.$$

Definition 3 (l_p-norm)**.** *Let w a vector of $\mathbb{R}^n$. The l_p-norm is the function $\|.\|_p$ such that*

$$\|w\|_p = \left(\sum_{i=0}^{n-1} |w_i|^p \right)^{\frac{1}{p}}.$$

The l_1-norm is called the Manhattan norm or taxicab norm.
The l_2-norm is called the Euclidean norm.
The $l_\infty-$norm is called the infinity norm or max norm, and is computed as $\|w\|_\infty = \max\{|w_i|, 0 \leq i < n\}$.

Definition 4 (Successive Minima)**.** *Let $\mathcal{L}$ a lattice and an integer i. The i^{th} successive minima, noted λ_i is the smallest real number such there exist i non zero linear independent vector $v_1, \ldots, v_i \in \mathcal{L}$ with $\|v_1\|, \ \ldots, \|v_i\| \leq \lambda_i$. $\lambda_{i,p}$ refers to the i^{th} successive minima regarding the l_p-norm.*

Theorem 1 (of Minkowski [11])**.** *For any $d-$dimensional lattice $\mathcal{L}$, $\lambda_{1,\infty} \leq det(\mathcal{L})^{\frac{1}{d}}$.*

Ajtai [1] demonstrated that using the Gaussian heuristic, one can predict that in a $d-$dimensional random lattice,

$$\lambda_{1,2} \approx \frac{\Gamma\left(\frac{d}{2}+1\right)}{\sqrt{\pi}} det(\mathcal{L})^{\frac{1}{d}}. \tag{1}$$

Definition 5 (γ-Gap Shortest Vector Problem ($GapSVP_\gamma$))**.** *Given a lattice $\mathcal{L}$ and a real number r, determine if $\lambda_1 \leq r$ or $\lambda_1 > \gamma r$.*

Definition 6 (γ-Unique Shortest Vector Problem ($USVP_\gamma$))**.** *Given a lattice $\mathcal{L}$ with $\gamma\lambda_1(\mathcal{L}) < \lambda_2(\mathcal{L})$, find $v \in \mathcal{L}$ such that $\|v\| = \lambda_1$. γ is called the gap.*

Definition 7 (α-Bounded Distance Decoding (BDD_α))**.** *Given a lattice $\mathcal{L}$ and a vector v such that $\exists u, (v-u) \in \mathcal{L}, \|u\| < \alpha\lambda_1(\mathcal{L})$, find u.*

2 Algorithm Specifications

2.1 Cryptosystem Concept

The cryptosystem requires three public parameters $\{d, b, p\}$ corresponding:

i) d a lattice dimension,

ii) b an upper bound,

iii) p a prime number.

Its trapdoor is a odd vector of bounded Manhattan norm i.e. a vector $w \in \mathcal{M}_{d,l}$,

$$\mathcal{M}_{d,l} = \{w \in \mathbb{Z}^d, \forall 1 \leq i \leq d, w_i \bmod 2 = 1, \|w\|_1 \leq l\}.$$

It could be described as follow:

- Setup: Alice creates a public $d-$dimensional lattice $\mathcal{L}$ of determinant p such there exists a vector $w \in \mathcal{L}^* \cap \in \mathcal{M}_{d,\frac{p-1}{2b}}$.
- Encryption: To encrypt a message $m = \{0,1\}$, Bob send to Alice a vector v such that $\exists u, v-u \in \mathcal{L}, \|u\|_\infty \leq b, \sum_{i=1}^{d} u_i \bmod 2 = m$.
- Decryption: To decrypt Alice uses w to extract the parity m of u from the vector v.

2.2 Specific Algorithm

To setup this cryptosystem, we require a uniform generator of vector of $\mathcal{M}_{d,l}$.

Algorithm 1 Random Generator from $\mathcal{M}_{d,l}$

```
Require: $d$ a dimension and $l$ a bound.
Ensure: $w \in \mathcal{M}_{d,l}$
1: Compute $l' = \lfloor \frac{l+d}{2} \rfloor$
2: Pick a random combination of $d-1$ elements $a_i$ in $[1, l']$ i.e. $1 \leq a_1 < ... < a_i < ... < a_d \leq l'$
3: Compute $w'$ such $w'_i = a_i - a_{i-1}$ for $i > 1$ and $w'_1 = a_1$.
4: Compute $w''$ such $w''_i = 2w'i - 1$.
5: Compute $w$ such $w_i = s_i w''_i$ with $s_i$ a random elements of $\{-1, 1\}$.
6: return $w$
```

Proof of Algorithm 1 correctness.
Line 2 and 3 corresponds to a known method ([13] paragraph 2.5.3, [15] for a specific integer version) to create random strictly positive vector of l_1-norm less or equal to l'. Line 4 transforms a strictly positive vector of l_1-norm less or equal to l' in a positive odd vectors of l_1-norm less or equal to $2l' - d$. As $l' = \lfloor \frac{l+d}{2} \rfloor = \frac{l+d-(l+d \bmod 2)}{2}$, we have $2l' - d = l - (l + d \bmod 2)$. Indeed, this is correct, as if $l + d = 1 \bmod 2$ i.e. l and d have different parity, there is no odd vectors of l_1-norm equal to l. Line 5 transforms a positive odd vectors to a signed odd vectors without changing the l_1-norm. This is uniform as well as all elements of $w_i \neq 0$. □

2.3 CPA encryption

Algorithm 2 Key Generation

```
Require: A set of parameters $\{d, b, p\}$.
Ensure: A secret key $SK = w_1$ with $w_1 = [1, p-1]$ and a public key $PK = v$ with $v \in [1, p-1]^d$.

Create a random vector of $w$ of $\mathcal{M}_{d, \lfloor \frac{p-1}{2b} \rfloor}$.
Compute $h \in \mathbb{Z}^d$ with $h_i = w_i(w_1^{-1}) \bmod p$.
Set $SK = w_1$ and $PK = h$.
return $SK, PK$.
```

Algorithm 3 Encryption

```
Require: A set of parameters $\{d, b, p\}$, a public key $PK = h$ and message $m = \{0, 1\}$.
Ensure: A ciphertext $CT = s$ with $s \in [0, p[$.
Create a random vector $u \in [-b, b]^d$ with $\sum u_i = 1 \bmod 2$.
Compute $s = \langle u, h \rangle \bmod p$.
Set $CT = s$.
return $CT$.
```

Algorithm 4 Decryption

```
Require: A set of parameters $\{d, b, p\}$, a secret key $SK = w_1$ and ciphertext $CT = s$.
Ensure: A message $m = \{0, 1\}$.
Compute $r = sw_1 \bmod p$ with $r \in ]-\frac{p}{2}, \frac{p}{2}[$.
Compute $m = r \bmod 2$
return $m$.
```

Proof of Algorithm 4 correctness.
We can check that

$$\begin{aligned} r &= sw_1 \bmod p \\ &= \langle u, h \rangle w_1 \bmod p \\ &= \langle u, hw_1 \rangle \bmod p \\ &= \langle u, ww_1^{-1} w_1 \rangle \bmod p \\ &= \langle u, w \rangle \bmod p \end{aligned}$$

Furthermore,

$$\begin{aligned} |\langle u, w \rangle| &\leq \|u\|_\infty \|w\|_1 \\ &\leq b\|w\|_1 \\ &\leq b\frac{p-1}{2b} \\ &\leq \frac{p-1}{2} \\ &< \frac{p}{2} \end{aligned}$$

Consequently, with $r = \langle u, w \rangle \bmod p$ and $-\frac{p}{2} < r < \frac{p}{2}$, we have $r = \langle u, w \rangle$.
Finally, we have

$$\begin{aligned} r \bmod 2 &= \langle u, w \rangle \bmod 2 \\ &= \sum_{i=1}^{d} u_i w_i \bmod 2 \\ &= \sum_{i=1}^{d} u_i 1 \bmod 2 \\ &= \sum_{i=1}^{d} u_i \bmod 2 \\ &= m \end{aligned}$$

□

2.4 CCA Key Encapsulation Message

We are using a method proposed by Dent [3] to transform a Chosen-Plaintext Attack (CPA) resistant cryptosystem into a Chosen-Ciphertext Attack (CCA) resistant Key Encapsulation Mechanism (KEM) of λbits security:

1. Alice sets the random generator used by Algorithm 3 with a seed s of λbits.

2. Alice uses Algorithm 3 to encrypt bit by bit s and send c, the ciphertext, to Bob,

3. Bob uses Algorithm 4 to extract the seed s from c,

4. Bob sets the random generator used by Algorithm 3 with s,

5. Bob uses Algorithm 3 to encrypt bit by bit s,

6. Bob checks that the ciphertext generated is equivalent to c,

7. Alice and Bob both expand s to a share secret.

3 Security Analysis

3.1 Public Key Security

Regarding public key security, a first remark is that giving h with $h_1 = 1$ to represent the lattice $\mathcal{L} = \{v \in \mathbb{Z}^d, \langle v, h\rangle = 0 \bmod p\}$ is equivalent of giving a basis B of $\mathcal{L}$,

$$B = \begin{pmatrix} p & 0 & 0 & \cdots & 0 & 0 \\ p-h_2 & 1 & 0 & \cdots & 0 & 0 \\ p-h_3 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ p-h_{d-1} & 0 & 0 & \cdots & 1 & 0 \\ p-h_d & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

We present Theorem 2 showing how random is the public key.

Theorem 2. *Let $\mathcal{L}$ a full rank lattice of determinant $p > 2$ prime and dimension $d > 1$, and $l \in \mathbb{N}^*$, the probability $\mathcal{P}_{p,d,l}$ that $\mathcal{L}^* \cap \mathcal{M}_{d,l} = \varnothing$ is given by*

$$\mathcal{P}_{p,d,l} = \left(1 - \frac{1}{p^{d-1}}\right)^{2^{d-1}\binom{\lfloor \frac{l+d}{2} \rfloor}{d}} \tag{2}$$

Proof of Theorem 2.

i) Firstly, we present the probability than an element of $\mathbb{Z}^d$ is in $\mathcal{L}^*$.
If $\mathcal{L}$ has a basis

$$\begin{pmatrix} p & 0 & 0 & \cdots & 0 & 0 \\ a_1 & 1 & 0 & \cdots & 0 & 0 \\ a_2 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ a_{d-1} & 0 & 0 & \cdots & 1 & 0 \\ a_d & 0 & 0 & \cdots & 0 & 1 \end{pmatrix},$$

then $\mathbb{Z}^d \cap \mathcal{L}^*$ has a basis given by

$$\begin{pmatrix} 1 & -a_1 & -a_2 & \cdots & -a_{d-1} & -a_d \\ 0 & p & 0 & \cdots & 0 & 0 \\ 0 & 0 & p & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & p & 0 \\ 0 & 0 & 0 & \cdots & 0 & p \end{pmatrix}$$

Therefore, the probability that a vector of $\mathbb{Z}^d$ is in $\mathbb{Z}^d \cap \mathcal{L}^*$ is given by $\frac{1}{p^{d-1}}$.

ii) Secondly, Algorithm 1 allows us to deduce the size of $\mathcal{M}_{d,l}$,

$$\#\mathcal{M}_{d,l} = 2^d \binom{\lfloor \frac{l+d}{2} \rfloor}{d}.$$

As if $v \in \mathbb{Z}^d \cap \mathcal{L}^*$ then $-v \in \mathbb{Z}^d \cap \mathcal{L}^*$, we restrict the number of relevant vectors to

$$2^{d-1}\binom{\lfloor \frac{l+d}{2} \rfloor}{d}$$

i.e. we study only vectors of $\mathcal{M}_{d,l}$ with a positive first coefficient. As p and 2 are coprime, each vectors of $\mathcal{M}_{d,l}$ with a positive coefficient has the same independant probability to be in $\mathcal{L}^*$.

□

To allow a better understanding, Equation 2 can be approximate

$$\begin{aligned} \mathcal{P}_{p,d,l} &\approx e^{-2^{d-1}\binom{\lfloor \frac{l+d}{2} \rfloor}{d} p^{-(d-1)}} \\ &\approx e^{-\binom{\lfloor \frac{l+d}{2} \rfloor}{d} \left(\frac{p}{2}\right)^{-(d-1)}} \end{aligned}$$

A relevant special case is when we fix $l = \frac{p-1}{2b}$, we can further approximate

$$\begin{aligned} \mathcal{P}_{p,d,\frac{p-1}{2b}} &\approx e^{-\binom{\lfloor \frac{l+d}{2} \rfloor}{d} \left(\frac{p}{2}\right)^{-(d-1)}} \\ &\approx e^{-\binom{\lfloor \frac{p-1+2db}{4b} \rfloor}{d} \left(\frac{p}{2}\right)^{-(d-1)}} \end{aligned}$$

Finally, assuming $p \gg b, d$, we can simplify further in

$$\begin{aligned} \mathcal{P}_{p,d,\frac{p-1}{2b}} &\approx e^{-\frac{\left(\frac{p}{4b}\right)^d}{d!} \left(\frac{p}{2}\right)^{-(d-1)}} \\ &\approx e^{-\frac{p}{2^{d+1} b^d (d)!}} \end{aligned}$$

Therefore, we obtain that more than half of lattice $\mathcal{L}$ with a determinant $p \gtrsim 2^{d+1} b^d (d)!$ has such a trapdoor, i.e. $\mathcal{M}_{d,\lfloor \frac{p-1}{2b} \rfloor} \cap \mathcal{L}^* \neq \varnothing$. It guarantees a highest security for our public key as any information from the public key will be as hard to extract as from a random lattice.

Consequently, we will use

$$2^{d+1} b^d (d)! < p < 2^{d+2} b^d (d)!. \tag{3}$$

3.2 Message Security

Message security is based on a new problem (Definition 8).

Definition 8 (α-Bounded Distance Parity Check (BDPCα))**.** *Given a lattice $\mathcal{L}$ of dimension d and a vector v such that $\exists u, (v-u) \in \mathcal{L}, \|u\| < \alpha\lambda_1(\mathcal{L})$, find $\sum_{i=1}^{d} u_i \bmod 2$.*

This problem is clearly related to a more studied problem i.e. the Bounded Distance Decoding problem (Definition 7). We present Theorem 3 to confirm this intuition.

Theorem 3 ($BDD_{\frac{\alpha}{4}} \leq BDPC_\alpha$)**.** *For any l_p−norm and any $\alpha \leq 1$ there is a polynomial time Cook-reduction from $BDD_{\frac{\alpha}{4}}$ to $BDPC_\alpha$.*

Proof of Theorem 3.
Let $\mathcal{L}, v$ and instance of $BDD_{\frac{\alpha}{4}}$. The main idea of the reduction is to use a $BDPC_\alpha$ solver to find the parity of one coefficient (instead of the sum of them), do it for each coefficient, use this information to transform v, u in both all even vectors and consequently divide them by 2. Repeating all this operation allows to extract u.

i) Find the parity of u_1.

Let B a basis of $\mathcal{L}$ and P a matrix defined by

$$D = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

We compute $v' = vD$ and $B' = BD$. As $v = u + kB$ with $k \in \mathbb{Z}^d$, we obtain $v' = uD + kB'$. We note $u' = uD$.

First, we note that $\|u'\| \leq 2\|u\|$ for any l_p-norm.

Secondly, we know that there is at least one non-null vector $w' \in \mathcal{L}'$ such that $\|w'\| = \lambda_1(\mathcal{L}')$. If we note $w = w'D^{-1}$ with

$$D^{-1} = \begin{pmatrix} 1 & -1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix},$$

we obtain a non-null vector w in $\mathcal{L}$ with $\|w\| \leq \|w'\|\|D^{-1}\| = \lambda_1(\mathcal{L}')\|D^{-1}\|$. Once again, for any l_p-norm, we have $\|w\| \leq 2\lambda_1(\mathcal{L}')$. As $\|w\|$ is a non null vector of $\mathcal{L}$, we can then obtain $\lambda_1(\mathcal{L}) \leq \|w\| \leq 2\lambda_1(\mathcal{L}')$.

By definition, we have $\|u\| < \frac{\alpha}{4}\lambda_1(\mathcal{L})$. Using the two equations, $\frac{\|u'\|}{2} \leq \|u\|$ and $\lambda_1(\mathcal{L}) \leq 2\lambda_1(\mathcal{L}')$, we obtain $\|u'\| < \alpha\lambda_1(\mathcal{L}')$.

Consequently, we can use a $BDPC_\alpha$ oracle respectively on $\mathcal{L}', v'$ and $\mathcal{L}, v$ and extract respectively $\sum_{i=1}^d u'_i \bmod 2$ and $\sum_{i=1}^d u_i \bmod 2$.

Substracting those two value, we obtain the parity of u_1, i.e.

$$\sum_{i=1}^d u'_i - \sum_{i=1}^d u_i \bmod 2 = u_1 + \sum_{i=1}^d u_i \bmod 2 - \sum_{i=1}^d u_i \bmod 2 = u_1 \bmod 2.$$

ii) Find the parity of u_i.

By using permutation matrix P with

$$P = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix},$$

we can repeat the first operation on BP and vP to obtain u_2. We can repeat this operation with equivalent type of permutation matrix to obtain all u_i.

iii) Dividing v by two.

From i and ii, we know we can extract z such that $z_i = \{0,1\}$ and $z_i = u_i \bmod 2$.

We compute a vector $q \in \{0,1\}^d$ satisfying $v + qB = z \bmod 2$. Definition itself of the problem guarantees such a solution exists.

We note that $v' = \frac{v+qB-z}{2}$ is such that it exist a $k' \in \mathbb{Z}^d$ such that $v' = \frac{u-z}{2} + k'B$ with $k' = \frac{k+q}{2}$.

We can now redo step i and ii with $v', \mathcal{L}$.

iv) Extracting u.

After s number of steps iii, we can rebuild from all z, Z such $u_i = Z_i \bmod 2^s$. Therefore, after

$$s = \left\lceil 1 + \log_2 \left(\alpha \left(det(\mathcal{L}) \right)^{\frac{1}{d}} \right) \right\rceil,$$

steps iii and using Theorem 1 and Definition 7, we obtain that

$$\begin{aligned} \|u\|_\infty &< \alpha \lambda_{1,\infty} \\ &< \alpha \left(det\left(\mathcal{L}\right)\right)^{\frac{1}{d}} \\ &< 2^{s-1} \end{aligned}$$

Consequently u can be extracted from Z as $u_i = Z_i \bmod 2^s$ with $-2^{s-1} < u_i < 2^{s-1}$.

□

Moreover, using Theorem 1 and Equation 3, we obtain that

$$\begin{aligned} \lambda_{1,\infty} &\le p^{\frac{1}{d}} \\ &< \left(2^{d+2} b^d (d)!\right)^{\frac{1}{d}} \\ &< 2b \left(4(d)!\right)^{\frac{1}{d}} \end{aligned}$$

As by definition $\|u\|_\infty = b$, we obtain that the message security is based on $\mathrm{BDPC}_{\alpha,\infty}$ with

$$\alpha \ge \frac{1}{2\left(4(d)!\right)^{\frac{1}{d}}}.$$

Using Theorem 3 as well as results form Lyubashevsky and Micciancio [10], we obtain a message security as hard as solving any of the following problems,

i) BDD_α with $\alpha = \frac{1}{o(d)}$ for $l_\infty-$norm,

ii) USVP_γ with $\gamma = o(d)$ for $l_\infty-$norm,

iii) GapSVP_γ with $\gamma = o(\frac{d^2}{\log d})$ for $l_\infty-$norm,

iv) GapSVP_γ with $\gamma = o(\frac{d^2}{\log d})$ for l_2-norm using a randomized reduction from [12].

Remark 2. *It is important to notice that weaker version of those problems are used in most of lattice based cryptography. This is indeed one of the strongest advantages of this cryptosystem. However, this has a cost: the size and time of key generation, encryption and decryption are somehow costly.*

3.3 Side Channel Attack Resistance

During encryption, to avoid a possible extraction via side channel of $\sum_{i=1}^{d} |u_i|$, we replace the computation of $\sum_{i=1}^{d} u_i h_i \bmod p$ by

$$\left(\sum_{i=1}^{d} (b+1+u_i)\, h_i\right) - (b+1)\sum_{i=1}^{d} h_i \bmod p.$$

Then, we can precompute for $1 \leq i \leq d, 1 \leq j \leq 2b+1$, $h_{i,j} = jh_{i,j} \bmod p$. Hence, the final computation becomes

$$\left(\sum_{i=1}^{d} h_{i,b+1+u_i}\right) - (b+1)\sum_{i=1}^{d} h_i \bmod p$$

which can be done in constant time.

3.4 Known Attacks

As shown in the previous section, the ciphertext is from far the easiest way to attack this cryptosystem. There is no specific algorithm to solve efficiently a BDD_∞ problem. As for lattice problem in non Euclidean norm, the most efficient attacks are made by using Euclidean norm solvers.

For u with $\|u\|_\infty \leq b$, we have u_i^2 to be expected as $\frac{2\sum_{j=1}^{b} j^2}{2b+1}$. Therefore, $\|u\|_2 \approx \sqrt{d\frac{(b+1)b}{2b+1}}$. Furthermore, using Gaussian Heuristic (Equation 1), one can expect to have $\lambda_{1,2} \approx \frac{\Gamma\left(\frac{d}{2}+1\right)^{\frac{1}{d}}}{\sqrt{\pi}} p^{\frac{1}{d}}$. Consequently, to successfully attack a message encrypted with parameters $\{d, b, p\}$, one need to solve an Euclidean norm BDD_α problem with

$$\alpha \approx \frac{\sqrt{\pi d\frac{(b+1)b}{2b+1}}}{\Gamma\left(\frac{d}{2}+1\right)^{\frac{1}{d}} p^{\frac{1}{d}}}.$$

However, even for the Euclidean norm, there are few results on solving efficiently a BDD problem [9].

Currently, the state of the art method would be

i) to transform a BDD problem into a Unique Shortest Vector Problem (Embedding Technique),

$$\begin{pmatrix} v & 1 \\ B & 0 \end{pmatrix},$$

ii) to solve this new USVP using lattice reduction algorithm.

Using this method, we obtain a USVP with a gap

$$\gamma \approx \frac{\Gamma\left(\frac{d+3}{2}\right)^{\frac{1}{d+1}} p^{\frac{1}{d+1}}}{\sqrt{\pi d\frac{(b+1)b}{2b+1}}}. \tag{4}$$

Lattice reduction methods are well studied and their strength are evaluated using the Hermite factor. Let $\mathcal{L}$ a $d-$dimensional lattice, the Hermite factor of a basis B of $\mathcal{L}$ is given by

$$\frac{\|B_1\|_2}{det(\mathcal{L})^{\frac{1}{d}}}.$$

Consequently, lattice reduction algorithms strengths are given by the Hermite factor of their expected output basis.

In [5], it was estimated that lattice reduction methods solve USVP$_\gamma$ with γ a fraction of the Hermite factor.

To confirm this hypothesis, we have performed lattice reduction on embedded BDD problem of encrypted message using LLL [8] and BKZ$_{20}$ [14]. Those lattice reduction algorithm have an expected Hermite factor of respectively 1.0219^d and 1.0128^d [5]. For each dimension d, we fix $b = 1$ and $p \approx 2^{d+1}b^d(d)! = 2^{d+1}(d)!$ as from Equation 3.

Figure 1 shows the success rate of the embedded attack using LLL and BKZ$_{20}$ in function of d the dimension. Tests have been performed on 48 different instances for each dimension and using FPLLL [4].

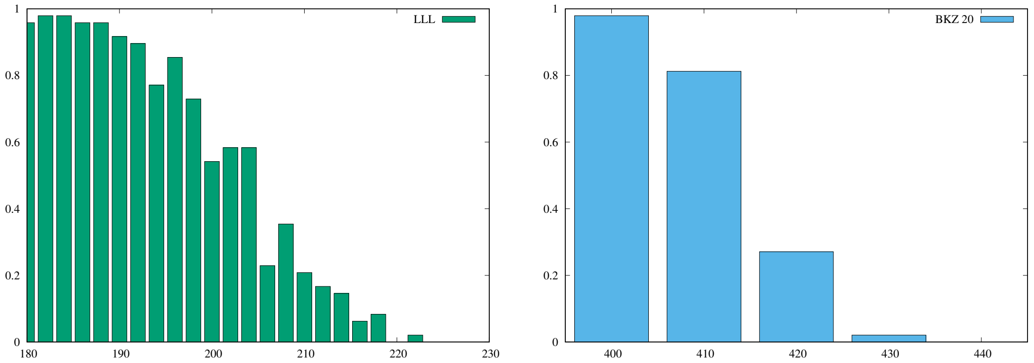

Figure 1: Success rate in function of lattice dimension.

Figure 2 shows the same results than in Figure 1, however, in function of the ratio estimated of the USVP gap to the estimated Hermite factor of lattice reduction algorithm. For LLL and BKZ$_{20}$, this ratio is given by $\frac{\Gamma(\frac{d+3}{2})^{\frac{1}{d+1}} p^{\frac{1}{d+1}}}{\sqrt{\pi\frac{2d}{3}}1.0219^{d+1}}$, $\frac{\Gamma(\frac{d+3}{2})^{\frac{1}{d+1}} p^{\frac{1}{d+1}}}{\sqrt{\pi\frac{2d}{3}}1.0128^{d+1}}$ respectively.

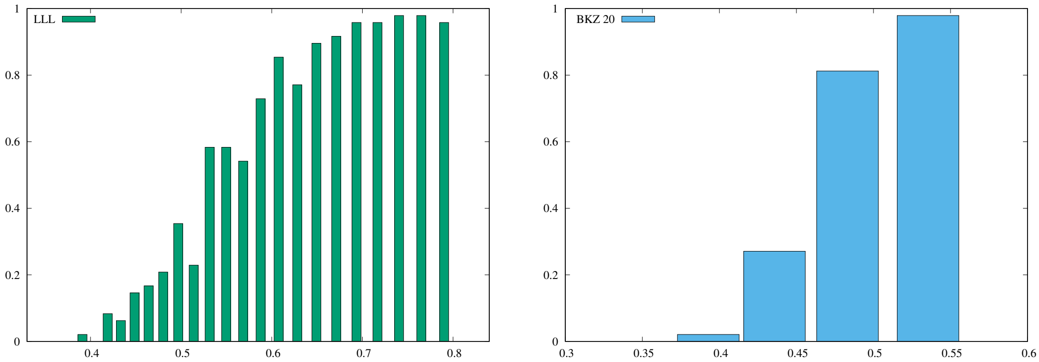

Figure 2: Success rate in function of the USVP gap to Hermite factor ratio.

Consequently, we will use a conservative bound of $\frac{1}{4}$ for the ratio of the USVP gap to the Hermite factor.

3.5 Expected Security Strength

Different papers are giving some relations between the Hermite factor and the security parameter λ [7, 17] often using BKZ simulation [2]. Aiming to be conservative, we are to assume a security of $2^{128}, 2^{192}, 2^{256}$ for a Hermite factor of $1.006^d, 1.005^d, 1.004^d$ respectively.

Dimension	Bound	Determinant	$\mathcal{P}_{p,d,\frac{p-1}{2b}}$	Gap	2^λ
965	1	$2^{9148} - 11919$	$\lesssim 0.235$	$< \frac{1}{4}(1.007)^{d+1}$	
1156	1	$2^{11258} - 4217$	$\lesssim 0.336$	$< \frac{1}{4}(1.006)^{d+1}$	2^{128}
1429	1	$2^{14353} - 15169$	$\lesssim 0.137$	$< \frac{1}{4}(1.005)^{d+1}$	2^{192}
1850	1	$2^{19268} - 7973$	$\lesssim 0.218$	$< \frac{1}{4}(1.004)^{d+1}$	2^{256}
2576	1	$2^{28057} - 6181$	$\lesssim 0.146$	$< \frac{1}{4}(1.003)^{d+1}$	

Table 1: Parameter Sets.

Remark 3. *The following choices have been made to build Table 1.*

i) p is set to respect equation 3,

ii) p is set to be a Pseudo Mersenne $p = 2^n - c$ which allows a compact representation n, c,

iii) b is set at 1 to obtain smaller keys and ciphertext size,

iv) d is set to obtain a USVP gap (Equation 4) with $\gamma < \frac{\delta^{d+1}}{4}$ for $\delta = 1.006, 1.005, 1.004$ respectively.

All those choices are not intrinsic to the cryptosystem and can be changed.

4 Implementation

4.1 Optimization

We have implemented two methods to accelerate time computation.

i) Pseudo Mersenne: as $p = 2^n - c$ with c small, we can accelerate modular reduction required during computation [16]. We replace $a \bmod (2^n - c)$ with $a_1 c + a_0 \bmod (2^n - c)$ when $a = a_0 + a_1 2^n$

ii) Computation Sharing: when encrypting λ bits, we can compute the encryption of all λ bits at the same time. This allows to share some resources. Indeed, instead of computing $\sum_{i=1}^{d} h_{i,u'_i}$ with $u'_i = b+1+u_i$ and $h_{i,j} = jh_i \bmod p$, one can precompute all possible $\sum_{l=0}^{k-1} h_{i+l,u'_{i+l}} \bmod p$ corresponding to all possible k consecutive additions of h_{i,u'_i} and memorize it in

$$H_{\sum_{l=0}^{k-1} u'_{i+l}(2b+1)^l}.$$

Therefore, if multiple ciphertexts require the same k consecutive additions, it is performed only one time. This method costs to memorize $(2b+1)^k$ values and is clearly limited as $(2b+1)^k < \lambda$ to allow a good number of share computations. In our tests, maximum efficiency has been obtained with $k = 3$ for $\lambda = 128$ and $\lambda = 192$, with $k = 4$ for $\lambda = 256$.

4.2 Performance Analysis

We fix D, the dimension d, B, the bound b, N, the number of bits of the determinant p, C, the constant such that $p = 2^N - C$, P, the number of bytes for p i.e. $P = \lceil \frac{N}{8} \rceil$. Then we can fix the number of bytes for the secret key to DP, the number of bytes for the public key to $(D-1)P$, and the number of bytes for the ciphertext to λP.

Table 2 shows size of secret and public key and ciphertext, and time of key generation, encryption and decryption (average on 50,000 instances) for different security parameters. Tests have been performed using a Linux 4.13.0-17 on a dual core Intel i7-6560U 2.20GHz (Turbo 3.20GHz) with 8GB of RAM.

	Size in Bytes			Time in Microseconds		
2^λ	Secret Key	Public Key	Ciphertext	Key Generation	Encryption	Decryption
2^{128}	1,627,648	1,626,240	180,224	45,696	16,317	17,701
2^{192}	2,565,055	2,563,260	344,640	74,486	31,558	34,590
2^{256}	4,456,650	4,454,241	616,704	134,801	64,375	70,592

Table 2: Size and Time performance.

Note if one desires a more compact secret key, one can simply keep the seed used during the key generation and rebuild the secret key at the begin of the decryption. However, we can see on Table 2 that this will increase by a factor 3 to 4 the decryption time.

4.3 Known Answer Tests

We use NIST Known Answer Tests generator to create our KAT. We picked a tiny dimension i.e. $d = 79$, a bound $b = 1$ and a prime determinant $p = 2^{469} - 283$.

References

[1] M. Ajtai. Generating random lattices according to the invariant distribution, 2006.

[2] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 1–20, 2011.

[3] Alexander W. Dent. A designer's guide to kems. In *Cryptography and Coding, 9th IMA International Conference, Cirencester, UK, December 16-18, 2003, Proceedings*, pages 133–151, 2003.

[4] The FPLLL development team. fplll, a lattice reduction library. Available at https://github.com/fplll/fplll, 2016.

[5] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, pages 31–51, 2008.

[6] Daniel Goldstein and Andrew Mayer. On the equidistribution of Hecke points. *Forum Mathematicum*, 15(2):165–189, 2003.

[7] Jeffrey Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, William Whyte, and Zhenfei Zhang. Choosing parameters for NTRUEncrypt. In *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, pages 3–18, 2017.

[8] A.K. Lenstra, H.W. Lenstra Jr., and L. Lovsz. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[9] Mingjie Liu and Phong Q. Nguyen. Solving BDD by enumeration: An update. In *Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, San Francisco,CA, USA, February 25-March 1, 2013. Proceedings*, pages 293–309, 2013.

[10] Vadim Lyubashevsky and Daniele Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 577–594, 2009.

[11] H. Minkowski. *Geometrie der Zahlen*. B. G. Teubner, Leipzig, 1896.

[12] Oded Regev and Ricky Rosen. Lattice problems and norm embeddings. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 447–456, 2006.

[13] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. John Wiley & Sons, Incorporated, October 2006.

[14] Claus-Peter Schnorr. Block reduced lattice bases and successive minima. *Combinatorics, Probability & Computing*, 3:507–522, 1994.

[15] Noah A. Smith and Roy W. Tromble. Sampling uniformly from the unit simplex, October 2004.

[16] Jerome Solinas. *Pseudo-Mersenne Prime*. Springer US, Boston, MA, 2005.

[17] Joop van de Pol and Nigel P. Smart. Estimating key sizes for high dimensional lattice-based systems. In *Cryptography and Coding - 14th IMA International Conference, IMACC 2013, Oxford, UK, December 17-19, 2013. Proceedings*, pages 290–303, 2013.