Problem Set #3

April 7, 2011 due April 14

A Special "Easy" Lattice

In this problem set we cover a few aspects of the special easy lattice of Micciancio-Peikert, which is used in their trapdoor construction. Below let n be the security parameter and let q be another parameter, polynomial in n. Denote $k = |q| = O(\log n)$ and let the binary representation of q be $q_{k-1} \dots q_1 q_0$, namely the q_i's are bits such that $q = \sum_{i=0}^{k-1} q_i 2^i$.

1 A Small Basis

A. Consider the vector $\vec{g} = \langle 1, 2, 4, \dots, 2^{k-1} \rangle \in \mathbb{Z}^k$, and the lattice $\Lambda^\perp(\vec{g}) = \{\vec{x} \in \mathbb{Z}^k : \langle \vec{g}, \vec{x} \rangle = 0 \pmod q\}$. Prove that the columns of the following matrix S_k form a basis for $\Lambda^\perp(\vec{g})$:

$$S_k \stackrel{\text{def}}{=} \begin{pmatrix} 2 & & & & & q_0 \\ -1 & 2 & & & & q_1 \\ & -1 & 2 & & & q_2 \\ & & \ddots & \ddots & & \\ & & & -1 & 2 & q_{k-2} \\ & & & & -1 & q_{k-1} \end{pmatrix} \tag{1}$$

B. Consider the $n \times nk$ matrix

$$G \stackrel{\text{def}}{=} \begin{pmatrix} -\vec{g}- & & & \\ & -\vec{g}- & & \\ & & \ddots & \\ & & & -\vec{g}- \end{pmatrix} \tag{2}$$

Describe a basis for the lattice $\Lambda^\perp(G) \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^{nk} : G\vec{x} = 0 \pmod q\}$. What is the determinant of this lattice?

2 Small Integer Solutions

A. For any $u \in \mathbb{Z}_q$, denote the u-coset of $\Lambda^\perp(\vec{g})$ by $\Lambda^\perp_u(\vec{g}) \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^k : \langle \vec{g}, \vec{x} \rangle = u \pmod q\}$. Describe a poly($n$)-time algorithm that given $u \in \mathbb{Z}_q$ outputs a vector $\vec{x} \in \Lambda^\perp_u(\vec{g})$ of length at most $\sqrt{k}$.

B. Recall that the discrete Gaussian distibution with parameter s over a lattice (or coset) $L \subset \mathbb{R}^d$, outputs each point $\vec{x} \in L$ with probability proportional to the Gaussian measure $\rho_s(\vec{x})$. Namely,

$$D_{L,s}(\vec{x}) \stackrel{\text{def}}{=} \frac{\rho_s(\vec{x})}{\rho_s(L)}, \quad \text{where } \rho_s(\vec{x}) \stackrel{\text{def}}{=} \exp\big(-\pi\|\vec{x}\|^2/s^2\big) \text{ and } \rho_s(L) = \sum_{\vec{u} \in L} \rho_s(\vec{u})$$

Describe a poly(n)-time algorithm that given $u \in \mathbb{Z}_q$ samples from the distribution $D_{\Lambda_u^\perp(\vec{g}),s}$, for a small parameter s. How small can you make s while still keeping the algorithm poly(n)-time? *Hint. you can use rejection sampling, and can use the fact that for any s one can efficiently sample from the discrete Gaussian distribution with parameter s over the integers $D_{\mathbb{Z}^k,s}$.*

C. Describe a poly(n)-time algorithm that given $\vec{u} \in \mathbb{Z}_q^n$ outputs a vector in $\Lambda_{\vec{u}}^\perp(G) \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^{nk} : G\vec{x} = \vec{u} \pmod q\}$ of size at most $\sqrt{nk}$ (for the matrix G from Equation 2). Also describe a poly(n)-time algorithm that given $\vec{u} \in \mathbb{Z}_q^n$ samples from $D_{\Lambda_{\vec{u}}^\perp(G),s}$, for a small parameter s.

3 Learning with Errors

A. Describe a poly(n)-time algorithm that solves the *learning with errors* problem with respect to $\vec{g}$. Namely, for a secret scalar s, the algorithm is given as input a vector $\vec{u} = s\vec{g} + \vec{e} \bmod q$ where $\vec{e}$ is a "small error vector" with entries smaller than $q/8$ in absolute value. Your algorithm needs to recover the secret s.

B. Describe a poly(n)-time algorithm that inverts the function $\mathrm{LWE}_G(\vec{s}, \vec{e}) = \vec{s}G + \vec{e} \bmod q$, where $\vec{s} \in \mathbb{Z}_q^n$ and $\vec{e} \in \big[-\left\lfloor\frac{q-1}{8}\right\rfloor, \left\lfloor\frac{q-1}{8}\right\rfloor\big]^{nk}$.

C. The purpose of this question is to show how to use a Micciancio-Peikert G-trapdoor to solve LWE with respect to an arbitrary matrix A. Below let $A_1 \in \mathbb{Z}_q^{n \times m_1}$ and $A_2 \in \mathbb{Z}_q^{n \times nk}$, and denote $A = [A_1|A_2] \in \mathbb{Z}_q^{n \times (m_1+nk)}$. Also let $R \in \{0, \pm 1\}^{m_1 \times nk}$ be such that $A_1R + A_2 = G \bmod q$. Describe a poly(n)-time algorithm that given the trapdoor R, inverts the function

$$\mathrm{LWE}_A(\vec{s}, \vec{e}) = \vec{s}A + \vec{e} \bmod q, \quad \text{where } \vec{s} \in \mathbb{Z}_q^n \text{ and } \ \vec{e} \in \Big[-\Big\lfloor\frac{q-1}{8(m_1+1)}\Big\rfloor, \Big\lfloor\frac{q-1}{8(m_1+1)}\Big\rfloor\Big]^{m_1+nk}.$$

Hint. For an input vector $\vec{u} = \mathrm{LWE}_A(\vec{s}, \vec{e}) \in \mathbb{Z}_q^{m_1+nk}$, consider the vector $\vec{v} = \vec{u} \cdot \begin{pmatrix} R \\ I \end{pmatrix} \bmod q$.