1 Regev’s cryptosystem

In Regev’s cryptosystem, we have an error term $e$ that is initialized according to a distribution $\chi$. In this homework, we investigate what happens, say due to a programmer error, $e$ is not properly randomized.

(a) Knowlets: Regev, CompLWE  ProblemID: RegevNoError
Time:
Difficulty:

We have a faulty implementation of Regev’s cryptosystem where $e = (0, \ldots, 0)$ always. The adversary gets the public-key $(A, b)$ and a ciphertext $(c_1, c_2)$. How can the adversary compute the plaintext? (Describe the computation steps performed by the adversary.)

Hint: If in doubt, first try to figure out how to solve the computational LWE problem (i.e., find $s$) when $e = 0$ always.

(b) Knowlets: Regev, CompLWE  ProblemID: RegevLittleError
Time:
Difficulty:

Now we have a slightly better implementation. $e$ now indeed contains some noise, but too little. In fact, it turns out that with probability close to 1, only one component $e_i \neq 0$. (That is, for all $j \neq i$, $e_j = 0$.) Show that this is too little noise by giving an attack. (Given public key and ciphertext find the plaintext. Describe the computation steps performed by the adversary.)

(c) Knowlets: Regev  ProblemID: RegevA0
Time:
Difficulty:

Now we have a different randomness failure. $e$ is chosen properly, but $A = 0$. How to attack? (Given public key and ciphertext find the plaintext. Describe the computation steps performed by the adversary.)

(d) Knowlets: Regev  ProblemID: RegevManyMsg
Time:
Difficulty:

Consider the following variant of Regev’s scheme:
• **Encryption.** To encrypt \( \mu \in \mathbb{Z}_q \), pick \( x \leftarrow \{0,1\}^m \). Let \( c_1 := A^T x \) and \( c_2 := x \cdot b + \mu \) (all calculated in \( \mathbb{Z}_q \)).

That is, we have optimized the scheme by allowing messages in \( \mathbb{Z}_q \) (i.e., not limited to a single bit). This is much more efficient. What is the problem with this change?

<table>
<thead>
<tr>
<th>Knowlets:</th>
<th>Regev</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProblemID:</td>
<td>RegevMall</td>
</tr>
<tr>
<td>Time:</td>
<td></td>
</tr>
<tr>
<td>Difficulty:</td>
<td></td>
</tr>
</tbody>
</table>

And now something completely different: Given a ciphertext \((c_1, c_2)\) that is the encryption of some unknown \( \mu \in \{0,1\} \), how to compute a ciphertext \((c'_1, c'_2)\) that decrypts to \(1 - \mu\) (with high probability)?

**Note:** You do not need to prove that your solution is correct, it is enough to specify the algorithm.

**Note:** What you are showing here is that Regev’s cryptosystem is malleable.