Key Recovery Attacks against NTRU-based Somewhat Homomorphic Encryption Schemes

Massimo Chenal and Qiang Tang

APSIA group, SnT, University of Luxembourg
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg
{massimo.chenal; qiang.tang}@uni.lu

Abstract. A key recovery attack allows an attacker to recover the private key of an underlying encryption scheme when given a number of decryption oracle accesses. Previous research has shown that most existing Somewhat Homomorphic Encryption (SHE) schemes suffer from this attack. In this paper, we propose efficient key recovery attacks against two NTRU-based SHE schemes, which have not gained much attention in the literature. One is published by Lopez-Alt et al. at STOC conference 2012 and the other is published by Bos et al. at the IMACC conference 2013. Parallel to our work, Dahab, Galbraith and Morais have also proposed similar attacks but only for specific parameter settings at ICITS conference 2015. In comparison, our attacks apply to all parameter settings and are more efficient than theirs.

Keywords: Somewhat Homomorphic Encryption, Key Recovery Attack, IND-CCA1 Security.

1 Introduction

In the literature, all Somewhat Homomorphic Encryption (SHE) schemes have been developed with the aim of being IND-CPA secure. In [Gen09], Gentry emphasized it as a future work to investigate SHE schemes with IND-CCA1 security (i.e. secure against a non-adaptive chosen-ciphertext attack). Up to now, the only scheme proven IND-CCA1 secure is that by Loftus et al. [LMSV12]. Most works in this direction focus on devising attacks against existing SHE schemes.

It has been shown that most existing SHE schemes suffer from key recovery attacks, which allow an attacker to recover the private key of an underlying encryption scheme when given a number of decryption oracle accesses. It is clear that a key recovery attack is stronger than a typical attack against IND-CCA1 security.

1.1 Related Works

Loftus et al. [LMSV12] showed key recovery attacks against SHE schemes from [Gen09,GH11]. Zhang et al. [ZPS12] presented an attack against the SHE scheme in [vDGHV10]. Chenal and Tang [CT14] presented key recovery attacks for all the schemes in [BV11b,BV11a,GSW13,Bra12,BGV12].

Previous analysis has not paid much attention to the NTRU-based SHE schemes. Two representative schemes in this line are those by Lopez-Alt, Tromer and Vaikuntanathan [LATV12] and Jos et al. [BLLN13]. Note that, instead of relying on the original NTRU scheme by Hoffstein, Pipher and Silverman [HPS98] (NTRUEncrypt), these schemes are based on a variant by Stehle and Steinfeld [SS10]. Parallel to our work in this paper, we noticed that Dahab, Galbraith and Morais [DGM15] constructed key recovery attacks for these schemes from [BLLN13,LATV12].

It is worth noticing that there was a similar line of research which focused on chosen ciphertext attacks on the original NTRUEncrypt. (NTRUEncrypt lacked a proof of security; only in [SS10] it has been shown how to modify NTRUEncrypt to reduce security to standard problems in ideal lattices.) In [JJ00], the authors present a chosen ciphertext attack on NTRUEncrypt that recovers the secret key with some probability. However, as it has been noticed in [HGNP$^+$03], these attacks use fake ciphertexts and can therefore be easily thwarted. The authors of [HGNP$^+$03] exploit a weakness of the NTRUEncrypt scheme, i.e. the fact that validly generated ciphertexts can fail to decrypt, in order to develop attacks which use these decryption failures to recover the private key. Other key-recovery chosen-ciphertext attacks, following the same line of work, have been developed in [GN07].

1.2 Our Contribution

The key recovery attacks by Dahab, Galbraith and Morais [DGM15] work for arbitrarily-tailored parameters for the LTV12 and BLLN13 SHE schemes. For example, they require $6(t^2+t) < q$ and $B^2 < \frac{q}{36t^2}$ while these conditions are not assumed in [LATV12,BLLN13]. In this paper, we present attacks that work for all parameter settings. Moreover, our attacks are more efficient than theirs, see the following table. Note that n is defined as an integer of power of 2, B is a bound on the coefficient size of error distribution and is much smaller than q, $t \geq 2$ is an integer that partially determines the message space size. More detailed definitions for these parameters can be found in the following sections.

	Our Attacks	Attacks from [DGM15]
[LATV12]	$\lfloor \log_2 B \rfloor + n$	$n \cdot \lceil \log_2 B \rceil + n$
[BLLN13] (t is odd)	$\lceil \log_2 (B/t) \rceil$	$n \cdot \lceil \log_2 B \rceil$
[BLLN13] (t is even but not 2)	$\lceil \log_2 (B/t) \rceil + n$	$n \cdot \lceil \log_2 B \rceil$
[BLLN13] ($t = 2$)	$\lceil \log_2 (B/t) \rceil + n$	$n \cdot \lceil \log_2 B \rceil + n$

1.3 Structure of the Paper

In Section 2, we recall some background on SHE schemes. In Section 3, we present our attack against the LTV12 SHE scheme. In Section 4, we present our attack against the BLLN13 SHE scheme. In Section 5, we conclude the paper.

2 Preliminary

Let $\mathbb{N}$ be the set of natural numbers, $\mathbb{Z}$ the ring of integers, $\mathbb{Q}$ the field of rational numbers, and $\mathbb{F}_q$ a finite field with q elements, where q is a power of a prime p. In particular, we will consider often $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z} = \mathbb{Z}_p$. If $r \in \mathbb{Z}_q$, we indicate as r^{-1} its inverse in $\mathbb{Z}_q$, i.e. that value such that $r^{-1} \cdot r = 1 \bmod q$. For a ring R and a (two-sided) ideal I of R, we consider the quotient ring R/I. For a given rational number $x \in \mathbb{Q}$, we let $\lfloor x \rceil$, $\lfloor x \rfloor$ and $\lceil x \rceil$ be respectively the rounding function, the floor function and the ceiling function. For a given integer $n \in \mathbb{N}$, $\lfloor n + 1/2 \rceil = n+1$. Of course, our attacks work also, with trivial modifications, in the case we define $\lfloor n + 1/2 \rceil = n$. To indicate that an element a is chosen uniformly at random from a set A we use notation $a \overset{\$}{\leftarrow} A$. For a set A, we let its cardinality be $|A|$. We denote the map that reduces an integer x modulo q and uniquely represents the result by an element in the interval $(-q/2, q/2]$ by $[\cdot]_q$. Therefore, we will consider the ring $\mathbb{Z}_q$ as $\mathbb{Z}_q := \{-\lfloor \frac{q}{2} \rfloor, -\lfloor \frac{q}{2} \rfloor + 1, \ldots, \lfloor \frac{q}{2} \rfloor\}$. We extend this map to polynomials in $\mathbb{Z}[X]$ and thus also to elements of R by applying it to their coefficients separately; given a polynomial $a(x) \in R$, we define the map

$$[\cdot]_q : R \to R, \quad a(x) = \sum_{i=0}^{n-1} a_i x^i \mapsto \sum_{i=0}^{n-1} [a_i]_q x^i$$

Unless otherwise specified, λ will always denote the security parameter. In the asymmetric schemes we are going to discuss, the secret key is denoted as sk, and the public key is pk.

The following definitions are adapted from [Gen09]. We only assume bit-by-bit public-key encryption, i.e. we only consider encryption schemes that are homomorphic with respect to boolean circuits consisting of gates for addition and multiplication mod 2. Extensions to bigger plaintext spaces and symmetric-key setting are straightforward, so that we skip it.

Definition 1 (Homomorphic Encryption). *A public key homomorphic encryption (HE) scheme is a set $\mathcal{E} = (\mathsf{KeyGen}_{\mathcal{E}}, \mathsf{Encrypt}_{\mathcal{E}}, \mathsf{Decrypt}_{\mathcal{E}}, \mathsf{Evaluate}_{\mathcal{E}})$ of four algorithms all of which must run in polynomial time. When the context is clear, we will often omit the index $\mathcal{E}$.*

$\mathsf{KeyGen}(\lambda) = (\mathsf{sk}, \mathsf{pk})$
- input: λ
- output: sk; pk

$\mathsf{Encrypt}(\mathsf{pk}, m) = c$
- input: pk and plaintext $m \in \mathbb{F}_2$
- output: ciphertext c

$\mathsf{Decrypt}(\mathsf{sk}, c) = m'$

- input: sk and ciphertext c
- output: $m' \in \mathbb{F}_2$

$\mathsf{Evaluate}(\mathsf{pk}, C, (c_1, \ldots, c_r)) = c_e$

- input: pk, a circuit C, ciphertexts $c_1, \ldots, c_r$, with $c_i = \mathsf{Encrypt}(\mathsf{pk}, m_i)$
- output: ciphertext c_e

Informally, a homomorphic encryption scheme that can perform only a limited number of $\mathsf{Evaluate}$ operations is called a Somewhat Homomorphic Encryption (SHE) scheme.

A public-key encryption scheme is IND-CCA1 secure if a polynomial time attacker can only win the following game with a negligible advantage $\mathrm{Adv}^{\text{IND-CCA1}}_{\mathcal{A},\mathcal{E},\lambda} = |\Pr(b = b') - 1/2|$.

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $(m_0, m_1) \leftarrow \mathcal{A}_1^{(\mathsf{Decrypt})}(\mathsf{pk})$ /* Stage 1 */
- $b \leftarrow \{0, 1\}$
- $c^* \leftarrow \mathsf{Encrypt}(m_b, \mathsf{pk})$
- $b' \leftarrow \mathcal{A}_2(c^*)$ /* Stage 2 */

According to the definition, in order to show that a scheme is not IND-CCA1 secure, we only need to show that an adversary can guess the bit b with a non-negligible advantage given access to the decryption oracle in Stage 1. In comparison, in a *key recovery attack*, an adversary can output the private key given access to the decryption oracle in Stage 1. Clearly, a key recovery attack is stronger and can result in more serious vulnerabilities in practice.

2.1 Impact of Key Recovery Attacks

In theory, IND-CPA security may be enough for us to construct cryptographic protocols, in particular if we assume semi-honest attackers. However, key recovery attacks will pose serious threat for practical usage of SHE schemes if an attacker becomes malicious (or, an honest party is compromised) and submits manipulated ciphertexts to observe the behavior of the decryptor. We illustrate this point by presenting an "attack" against the LWE-based single-server private information retrieval (PIR) protocol in [BV11b].

The PIR protocol is very simple: the client has a long-term key tuple for a SHE scheme and a secret key sk for a symmetric encryption scheme; a PIR query is an encrypted index under sk; a PIR response is a ciphertext under the SHE public key, generated by the server (who is given the ciphertext of sk under the SHE public key) by homomorphically evaluating the encrypted index and the database; the client obtains the desired bit by decrypting the ciphertext using the SHE private key.

Clearly, if the server is malicious, then it can mount a key recovery attack by manipulating the responses and monitoring the client's behavior. With the SHE private key, the server can recover all the private information of the client. In order to prevent the attack, the client can require the server to prove all computations are done properly. However, this might make the server's computational complexity very heavy and make the protocol less efficient than others.

3 Attack against the LTV12 SHE Scheme

We start by recalling the LTV12 SHE Scheme [LATV12]. Let λ be the security parameter, consider an integer $n = n(\lambda)$ and a prime number $q = q(\lambda) \neq 2$. Consider also a degree-n polynomial $\phi(x) = \phi_\lambda(x)$: following [LATV12], we will use $\phi(x) = x^n + 1$. Finally, let $\chi = \chi(\lambda)$ a $B(\lambda)$-bounded error distribution over the ring $R := \mathbb{Z}[x]/(\phi(x))$. The parameters $n, q, \phi(x)$ and χ are public and we assume that given λ, there are polynomial-time algorithms that output n, q and $\phi(x)$, and sample from the error distribution χ. The message space is $\mathcal{M} = \{0, 1\}$, and all operations on ciphertexts are carried out in the ring $R_q := \mathbb{Z}_q[x]/(\phi(x))$.

$\mathsf{KeyGen}(\lambda)$:

- sample $f', g \leftarrow \chi$
- set $f := 2f' + 1$ so that $f \equiv 1 \bmod 2$
- if f is not invertible in R_q, resample f'
- $\mathsf{pk} := h = 2gf^{-1} \in R_q$
- $\mathsf{sk} := f \in R$

$\mathsf{Encrypt}(\mathsf{pk}, m)$:
- sample $s, e \leftarrow \chi$
- output ciphertext $c := hs + 2e + m \in R_q$

$\mathsf{Decrypt}(\mathsf{sk}, c)$:
- let $\mu = f \cdot c \in R_q$
- output $\mu' := \mu \bmod 2$

Since we don't need the evaluation step, we omit it in the description. In the original paper [LATV12], the somewhat homomorphic encryption scheme is multi-key, i.e. one can use several secret keys $\mathsf{sk}_1 = f_1, \ldots, \mathsf{sk}_M = f_M$ in order to decrypt. By analyzing the original decryption step, one can see that, in order to decrypt the plaintext message, we need to multiply secret keys $\mathsf{sk}_1 = f_1, \ldots, \mathsf{sk}_M = f_M$ together, and then multiply the result with the ciphertext and reduce. For this reason, it is enough to retrieve, as the secret key, the polynomial $f_1 \cdots f_M =: s = s(x) = s_0 + s_1 x + s_2 x^2 + \cdots + s_{n-1} x^{n-1} \in R_q$, with $s_i \in (-q/2, q/2]$ for all $i = 0, 1, \ldots, n-1$. For this reason, it is enough to present the scheme as we saw it, with only one secret key.

Remark 1. In [LATV12], the authors do not explicitly state how the decryption behaves if $\mu \bmod 2$ is not a constant. We consider three scenarios: (1) output directly $\mu \bmod 2$; (2) output the constant of $\mu \bmod R_2$; (3) output an error. In the following, we describe a key recovery attack for scenario (1) and it can be easily extended to scenario (2). It is likely that we can adapt our attack to scenario (3), but we have not succeeded so far.

3.1 Attack Preview

Generally, suppose the secret key is in the form of the polynomial $f = s(x) = s_0 + s_1 x + s_2 x^2 + \cdots + s_{n-1} x^{n-1} \in R_q$. Now, since we assume q odd, and s_i is an integer, we have $-q/2 < s_i < q/2$, and in particular $-\left\lfloor \frac{q}{2} \right\rfloor \leq s_i \leq \left\lfloor \frac{q}{2} \right\rfloor, \quad \forall 0 \leq i \leq n-1$. Each coefficient s_i can have $\left\lfloor \frac{q}{2} \right\rfloor - (-\left\lfloor \frac{q}{2} \right\rfloor) + 1 = q$ possible different values. We remark that there exists a bit representation of the s_i's such that $\#\text{bits}(s_i) = \lfloor \log_2(q-1) \rfloor + 1 =: N$, and $\#\text{bits}(s) = n \cdot \#\text{bits}(s_i) = n \cdot (\lfloor \log_2(q-1) \rfloor + 1)$. The decryption oracle reveals a polynomial $\mu'(x) = \mu(x) \bmod 2 = \mu'_0 + \mu'_1 x + \cdots + \mu'_{n-1} x^{n-1}$, with $\mu'_i \in \{0, 1\}$ for $i = 0, 1, \ldots, n-1$. Hence, decryption oracle reveals n bits at a time. Therefore, the minimum number of oracle queries needed to recover s is N. As we will see, our attack needs N oracle queries, plus at most $n-1$ oracle queries necessary to determine the signs of the coefficients of the secret key. We remark that the scheme as described in [LATV12] has message space $\mathcal{M} = \{0, 1\}$. When the oracle decryption receives an honestly-generated ciphertext, it returns either $0 = \sum_{i=0}^{n-1} 0 \cdot x^i \in R_q$ or $1 = 1 + \sum_{i=1}^{n-1} 0 \cdot x^i \in R_q$. However, in principle the oracle decryption can return any polynomial in $\{0, 1\}/(x^n + 1)$ and we will use this fact as basis to build our attack.

Here is the workflow of our key recovery attack. First of all, we are going to determine the parity of each coefficient $s_i \in (-q/2, q/2]$. Then, we are going to find s_i by gradually reducing (halving) the interval in which it lies. At some point, s_i will be reduced to belong to some interval with at most two consecutive integers; the absolute value of s_i will be deduced by its (known) parity. At this point, we will know the secret key coefficient s_i in absolute value; in the last step, we are going to query the oracle decryption at most n times in order to recover the sign of the coefficients s_i, for $i = 1, 2, \ldots, n-1$, relative to the (unknown) sign of s_0. So in the end, we will end up with two possible candidate secret keys $s_1(x)$ and $s_2(x) = -s_1(x)$. We have then $s(x) = s_1(x)$ or $s(x) = s_2(x)$, and recovering which one of the two is trivial with an extra oracle query.

In our description, we consider the coefficients s_i in the interval $(-q/2, q/2]$ and can recover the private key with at most $\lfloor log_2 q \rfloor + n$ decryption oracle queries. However, we could consider the stricter interval $[-B, B]$, with B the bound on coefficients given by the distribution χ from which the coefficients are picked from. In this case, we can see that the total number of queries needed to be submitted to the decryption oracle are actually at most $\lfloor log_2 B \rfloor + n$.

3.2 Detailed Attack

Preliminary Step.

Submit to the decryption oracle the "ciphertext" $c(x) = 1 \in R_q$. The oracle will compute and return the polynomial $D(c(x) = 1) = s(x) \bmod 2 = \sum_{i=0}^{n-1} (s_i \bmod 2) x^i$, which tells us the parity of each s_i, $i = 0, 1, \ldots, n-1$.

Step 1.
Choose and submit to the decryption oracle the "ciphertext" $c(x) = 2 \in R_q$. It will compute and return the polynomial $D(c(x) = 2) = (2s(x) \in R_q) \bmod 2 = \sum_{i=0}^{n-1} [(2s_i \bmod q) \bmod 2] \, x^i$. For all $i \in [0, n-1]$ we have

$$\frac{-q+1}{2} \leq s_i \leq \frac{q-1}{2}, \text{ and so } -q+1 \leq 2s_i \leq q-1 \tag{A}$$

For each i, we have two cases to distinguish:

Case A_1: $(2s_i \bmod q) \bmod 2 = 0$. Then, condition (A) implies that $\frac{-q+1}{2} \leq 2s_i \leq \frac{q-1}{2}$, i.e. $\frac{-q+1}{4} \leq s_i \leq \frac{q-1}{4}$

$$-q+1 \leq 4s_i \leq q-1 \tag{A1}$$

Case B_1: $(2s_i \bmod q) \bmod 2 = 1$. Then, condition (A) implies that $\frac{q-1}{2} + 1 \leq 2|s_i| \leq q-1$, i.e. $\frac{q+1}{4} \leq |s_i| \leq \frac{q-1}{2}$

$$q+1 \leq 4|s_i| \leq 2q-2 \tag{B1}$$

Step 2.
Choose and submit to the decryption oracle the "ciphertext" $c(x) = 4 \in R_q$. It will compute and return the polynomial $D(c(x) = 4) = [s(x) \cdot 4]_q \bmod 2 = \sum_{i=0}^{n-1} [[4s_i]_q \bmod 2] \, x^i$. For each i, we have four cases to distinguish:

Case A_2: In Step 1 case A_1 held, and $[4s_i]_q \bmod 2 = 0$. Then, condition (A1) implies that $\frac{-q+1}{2} \leq 4s_i \leq \frac{q-1}{2}$, i.e. $\frac{-q+1}{8} \leq s_i \leq \frac{q-1}{8}$

$$-q+1 \leq 8s_i \leq q-1 \tag{A2}$$

Case B_2: In Step 1 case A_1 held, and $[4s_i]_q \bmod 2 = 1$. Then, condition (A1) implies that $\frac{q-1}{2} + 1 \leq 4|s_i| \leq q-1$, i.e. $\frac{q+1}{8} \leq |s_i| \leq \frac{q-1}{4}$

$$q+1 \leq 8|s_i| \leq 2q-2 \tag{B2}$$

Case C_2: In Step 1 case B_1 held, and $[4s_i]_q \bmod 2 = 0$. Then, condition (B1) implies that $q+1+\frac{q-1}{2} \leq 4|s_i| \leq 2q-2$, i.e. $\frac{3q+1}{8} \leq |s_i| \leq \frac{q-1}{2}$

$$3q+1 \leq 8|s_i| \leq 4q-4 \tag{C2}$$

Case D_2: In Step 1 case B_1 held, and $[4s_i]_q \bmod 2 = 1$. Then, condition (B1) implies that $q+1 \leq 4|s_i| \leq \frac{3q-1}{2}$, i.e. $\frac{q+1}{4} \leq |s_i| \leq \frac{3q-1}{8}$

$$2q+2 \leq 8|s_i| \leq 3q-1 \tag{D2}$$

Step 3.
Choose and submit to the decryption oracle the "ciphertext" $c(x) = 8 \in R_q$. It will compute and return the polynomial $D(c(x) = 8) = [s(x) \cdot 8]_q \bmod 2 = \sum_{i=0}^{n-1} [[8s_i]_q \bmod 2] \, x^i$. For each i, we have four cases to distinguish:

Case A_3: In Step 2 case A_2 held, and $[8s_i]_q \bmod 2 = 0$. Then, condition (A2) implies that $\frac{-q+1}{2} \leq 8s_i \leq \frac{q-1}{2}$, i.e. $\frac{-q+1}{16} \leq s_i \leq \frac{q-1}{16}$

$$-q+1 \leq 16s_i \leq q-1 \tag{A3}$$

Case B_3: In Step 2 case A_2 held, and $[8s_i]_q \bmod 2 = 1$. Then, condition (A2) implies that $\frac{q-1}{2} + 1 \leq 8|s_i| \leq q-1$, i.e. $\frac{q+1}{16} \leq |s_i| \leq \frac{q-1}{8}$

$$q+1 \leq 16|s_i| \leq 2q-2 \tag{B3}$$

Case C_3: In Step 2 case B_2 held, and $[8s_i]_q \bmod 2 = 0$. Then, condition (B2) implies that $\frac{3q+1}{2} \leq 8|s_i| \leq 2q-2$, i.e. $\frac{3q+1}{16} \leq |s_i| \leq \frac{q-1}{4}$

$$3q+1 \leq 16|s_i| \leq 4q-4 \tag{C3}$$

Case D_3: In Step 2 case B_2 held, and $[8s_i]_q \bmod 2 = 1$. Then, condition (B2) implies that $q+1 \leq 8|s_i| \leq \frac{3q-1}{2}$, i.e. $\frac{q+1}{8} \leq |s_i| \leq \frac{3q-1}{16}$

$$2q+2 \leq 16|s_i| \leq 3q-1 \tag{D3}$$

Case E_3: In Step 2 case C_2 held, and $[8s_i]_q \bmod 2 = 0$. Then, condition (C2) implies that $\frac{7q+1}{2} \leq 8|s_i| \leq 4q-4$, i.e. $\frac{7q+1}{16} \leq |s_i| \leq \frac{q-1}{2}$

$$7q+1 \leq 16|s_i| \leq 8q-8 \tag{E3}$$

Case F_3: In Step 2 case C_2 held, and $[8s_i]_q \bmod 2 = 1$. Then, condition (C2) implies that $3q+1 \leq 8|s_i| \leq \frac{7q-1}{2}$, i.e. $\frac{3q+1}{8} \leq |s_i| \leq \frac{7q-1}{16}$

$$6q+2 \leq 16|s_i| \leq 7q-1 \tag{F3}$$

Case G_3: In Step 2 case D_2 held, and $[8s_i]_q \bmod 2 = 0$. Then, condition (D2) implies that $2q+2 \leq 8|s_i| \leq \frac{5q-1}{2}$, i.e. $\frac{q+1}{4} \leq |s_i| \leq \frac{5q-1}{16}$

$$4q+4 \leq 16|s_i| \leq 5q-1 \tag{G3}$$

Case H_3: In Step 2 case D_2 held, and $[8s_i]_q \bmod 2 = 1$. Then, condition (D2) implies that $\frac{5q+1}{2} \leq 8|s_i| \leq 3q-1$, i.e. $\frac{5q+1}{16} \leq |s_i| \leq \frac{3q-1}{8}$

$$5q+1 \leq 16|s_i| \leq 6q-2 \tag{H3}$$

Final step.

We continue in this fashion and finally we obtain integers $s'_i := |s_i| \in [0, \frac{q-1}{2}]$, for $i = 0, 1, \ldots, n-1$. This is obtained in the last step, where all coefficients $|s_i|$, in absolute value, can assume at most only two (consecutive) values; the known parity will then determine $|s_i|$. It is easy to see that in order to achieve this we need $\lfloor \log_2 q \rfloor$ steps.

The strategy now is to find out whether $s_i \cdot s_j < 0$ or $s_i \cdot s_j > 0$ holds, for every i, j with $s_i, s_j \neq 0$. Let s_m be the first non-zero coefficient. This way, we will obtain two possible candidates of the secret key, one with $s_m > 0$ and the other with $s_m < 0$. A trivial query to the oracle decryption will allow us to determine which is the correct secret key.

We have to choose an appropriate "ciphertext" $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ to submit to the decryption oracle. Choose $c_0 = 1, c_1 = 1$ and $c_j = 0$ for $j \neq 0, 1$. Oracle decryption will compute and return the polynomial

$$D(c(x)) = s(x) \cdot c(x) = [s_0 - s_{n-1}]_q \bmod 2 + \sum_{i=1}^{n-1}([s_i + s_{i-1}]_q \bmod 2)x^i$$

Fix $i = 1, 2, \ldots, n-1$ such that $s_i, s_{i-1} \neq 0$. Let $b_i := [s_i + s_{i-1}]_q \bmod 2$ be the coefficient of x^i, and let $b'_i := [s'_i + s'_{i-1}]_q \bmod 2$. There are two cases to consider:

- $s'_i + s'_{i-1} \geq \frac{q+1}{2}$. Then
  - if $b_i = b'_i$, then s_i and s_{i-1} have the same sign;
  - if $b_i \neq b'_i$, then s_i and s_{i-1} have different signs.
- $0 \leq s'_i + s'_{i-1} \leq \frac{q-1}{2}$. Then we need to make an extra query to understand whether s_i and s_{i-1} have the same sign or not.

Now, for each one of the i of the previous case (i.e. such that $0 \leq s'_i + s'_{i-1} \leq \frac{q-1}{2}$, $i = 1, 2, \ldots, n-1$, and $s_i, s_{i-1} \neq 0$) we choose and submit to the decryption oracle the polynomial $c(x) = \alpha_i |s_{i-1}| + \alpha_i |s_i| x$, i.e. we choose $c_0 = \alpha_i |s_{i-1}|$, $c_1 = \alpha_i |s_i|$, $c_2 = c_3 = \cdots = c_{n-1} = 0$, where α_i is chosen such that

$$\alpha_i |s_{i-1} \cdot s_i| \in \left(\frac{q-1}{4}, \frac{q-1}{2} \right] \tag{1}$$

(it is always possible to find such an α_i). The oracle decryption will return the polynomial

$$D(c(x)) = s(x) \cdot c(x) = [\alpha_i |s_{i-1}| s_0 - \alpha_i |s_i| s_{n-1}]_q \bmod 2 + \sum_{j=1}^{n-1} \left([\alpha_i |s_{i-1}| s_j + \alpha_i |s_i| s_{j-1}]_q \bmod 2 \right) x^j$$

Let's focus on the coefficient of x^i, i.e. $\beta_i := [\alpha_i |s_{i-1}| s_i + \alpha_i |s_i| s_{i-1}]_q \bmod 2$. Now, there are two cases:

- if s_i, s_{i-1} have different signs, then $\beta_i = 0$;
- if s_i, s_{i-1} have the same sign, then $\beta_i = 1$ (trivial to verify: 1 holds, and therefore $[2\alpha_i \cdot |s_i \cdot s_{i-1}|]_q$) is odd.

By repeating this idea for every $i = 1, 2, \ldots, n-1$ such that $0 \leq s'_i + s'_{i-1} \leq \frac{q-1}{2}$ we will know which one of the following relations $s_i \cdot s_{i-1} < 0 \quad \vee \quad s_i \cdot s_{i-1} > 0$ holds, for every consecutive non-zero coefficients s_i, s_{i-1}.

Now, we have one more thing to consider: we have to be careful in case one of the coefficient s_i is zero. In this case in fact, no information can be given about the sign of s_{i-1} if we compare it to s_i. To solve this problem, we have to choose and submit to the decryption oracle a polynomial $c(x) = a + bx^j$ for appropriates a, b, j. Let $0 \leq m_1 \leq n-1$ be an integer such that s_{m_1} is the first non-zero coefficient of the secret key $s(x)$. If there exists $i_1 > m_1$ such that $s_{i_1} = 0$, then let m_2 be the first non-zero coefficient such that $i_1 < m_2 \leq n-1$. Then we want to compare the relative signs of s_{m_1} and s_{m_2} by choosing the polynomial $c(x)$ with $c_0 = \alpha |s_{m_1}|$, $c_{m_2 - m_1} = \alpha |s_{m_2}|$, $c_j = 0$ for $j \neq 0, m_2 - m_1$. So we have $c(x) = \alpha |s_{m_1}| + \alpha |s_{m_2}| x^{m_2 - m_1}$, with α such that $\alpha |s_{m_1} s_{m_2}| \in \left(\frac{q-1}{4}, \frac{q-1}{2} \right]$. The oracle decryption will return the polynomial $D(c(x)) = s(x) \cdot c(x) = \beta_0 + \beta_1 x + \cdots + \beta_{n-1} x^{n-1}$. Consider the m_2-th coefficient $\beta_{m_2} = [\alpha |s_{m_1}| s_{m_2} + \alpha |s_{m_2}| s_{m_1}]_q \bmod 2$. As before, we can conclude that if s_{m_1}, s_{m_2} have different signs, then $\beta_{m_2} = 0$, and if s_{m_1}, s_{m_2} have the same sign, then $\beta_{m_2} = 1$.

Now, similar to what just discussed, if there exists $i_2 > m_2$ such that $s_{i_2} = 0$, then let m_3 be the first non-zero coefficient such that $m_3 > i_2$. We will in a similar fashion compare the relative signs of s_{m_1} and s_{m_3}. We keep proceeding this way, and in the end we will know, for every $0 \leq i, j \leq n-1$ such that $s_i \neq 0, s_j \neq 0$, whether $s_i \cdot s_j > 0$ or $s_i \cdot s_j < 0$ occurs. This allows us to determine two possible candidates for the secret key $s(x)$ (assume s_m is the first non-zero coefficient; then one candidate has $s_m < 0$, the other has $s_m > 0$). A trivial oracle decryption query will reveal which one of the two is the correct secret key. The total number of queries needed to be submitted to the oracle decryption query is then at most $\lfloor \log_2 q \rfloor + n$.

4 Attack against the BLLN13 SHE Scheme

We start by recalling the BLLN13 SHE Scheme [BLLN13]. For a given positive integer $d \in \mathbb{N}_{>0}$, define the quotient ring $R := \mathbb{Z}[x]/(\Phi_d(x))$, i.e. the ring of polynomials with integer coefficients modulo the d-th cyclotomic polynomial $\Phi_d(x) \in \mathbb{Z}[x]$. The degree of Φ_d is $n = \varphi(d)$, where φ is Euler's totient function. As considered by the authors of [BLLN13], for correctness of the scheme, let d be a power of 2; in this case, we have $\Phi_d(x) = x^n + 1$ with n also a power of 2. Therefore $R = \mathbb{Z}[x]/(x^n + 1)$. The other parameters of the [BLLN13] SHE scheme are a prime integer $q \in \mathbb{N}$ and an integer $t \in \mathbb{N}$ such that $1 < t < q$. Let also $\chi_{\text{key}}, \chi_{\text{err}}$ be two distributions on R. The parameters $d, q, t, \chi_{\text{key}}$ and χ_{err} are public and we assume that given λ, there are polynomial-time algorithms that output d, q, t and $\phi(x)$, and sample from the error distributions χ. The message space is $\mathcal{M} = R/tR = \mathbb{Z}_t[x]/(x^n + 1)$, and all operations on ciphertexts are carried out in the ring $R_q := \mathbb{Z}_q[x]/(\phi(x))$.

$\mathsf{KeyGen}(\lambda)$:

- sample $f', g \leftarrow \chi_{\text{key}}$
- let $f = [tf' + 1]_q$
- if f is not invertible in R_q, resample f'
- set $\mathsf{pk} := h = [tgf^{-1}]_q \in R_q$
- set $\mathsf{sk} := f \in R_q$

$\mathsf{Encrypt}(\mathsf{pk}, m)$**:**

- for a message $m + tR$, choose $[m]_t$ as its representative
- sample $s, e \leftarrow \chi_{\text{err}}$
- output ciphertext $c = [\lfloor q/t \rfloor [m]_t + e + hs]_q \in R_q$

$\mathsf{Decrypt}(\mathsf{sk}, c)$**:**

- output $m = \left[\left\lfloor \frac{t}{q} \cdot [fc]_q \right\rceil\right]_t \in R_t$

Since we don't need the evaluation step, we omit it in the description.

4.1 Attack Preview

We are going to recover the secret key $f(x) = f_0 + f_1 x + f_2 x^2 + \cdots + f_{n-1}x^{n-1} \in \frac{\mathbb{Z}_q[x]}{(x^n+1)}$, where f_i is an integer in $(-q/2, q/2]$ for all $i = 0, 1, \ldots, n-1$. In order to recover $f(x)$, we are going to submit specifically-chosen 'ciphertexts' of the form $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1}x^{n-1} \in \frac{\mathbb{Z}_q[x]}{(x^n+1)}$, with integers $c_i \in (-q/2, q/2]$. Choose $c(x) = 1 = 1 + 0x + 0x^2 + \cdots + 0x^{n-1}$. We have

$$\begin{aligned} D(c = 1) &= \left[\left\lfloor \frac{t}{q} \cdot [f \cdot 1]_q \right\rceil\right]_t = \left[\left\lfloor \frac{t}{q} \cdot \left([f_0]_q + [f_1]_q x + [f_2]_q x^2 + \cdots + [f_{n-1}]_q x^{n-1}\right) \right\rceil\right]_t \\ &\overset{*}{=} \left[\left\lfloor \frac{t}{q} \cdot \left(f_0 + f_1 x + \cdots + f_{n-1}x^{n-1}\right) \right\rceil\right]_t = \left[\left\lfloor \frac{t}{q} f_0 \right\rceil + \left\lfloor \frac{t}{q} f_1 \right\rceil x + \cdots + \left\lfloor \frac{t}{q} f_{n-1} \right\rceil x^{n-1}\right]_t \end{aligned}$$

Equality $\overset{*}{=}$ holds since the integer coefficients f_i are already reduced modulo q. Now, for every $0 \le i \le n-1$ we have $-q/2 < f_i \le q/2$. We have that $q > 2$ since in [BLLN13] it is claimed that $1 < t < q$, with t, q integers. In particular, q is a prime integer greater than 2, and therefore $q/2 \notin \mathbb{N}$. So we have $-q/2 < f_i < q/2$. In particular we have that $-\frac{t}{2} < \frac{t}{q} \cdot f_i < \frac{t}{2}$. For every $0 \le i \le n-1$, let $u_i^{(1)} := \left\lfloor \frac{t}{q} f_i \right\rceil$. We have $\lceil -\frac{t}{2} \rceil \le u_i^{(1)} \le \lfloor \frac{t}{2} \rfloor$. Each $u_i^{(1)}$ can have

$$\left\lfloor \frac{t}{2} \right\rfloor - \left\lceil -\frac{t}{2} \right\rceil + 1 = 2\left\lfloor \frac{t}{2} \right\rfloor + 1 = \begin{cases} t & \text{if } t \text{ is odd} \\ t+1 & \text{if } t \text{ is even} \end{cases}$$

possible different values, i.e. $u_i^{(1)}$ can have t different possible values if t is odd, and can have $t+1$ different possible values if t is even. Now, for every $0 \le i \le n-1$, we have that $[u_i^{(1)}]_t \in (-t/2, t/2]$ and therefore

- $[u_i^{(1)}]_t \in= [-\frac{t}{2} + \frac{1}{2}, -\frac{t}{2} + \frac{3}{2}, -\frac{t}{2} + \frac{5}{2}, \cdots, \frac{t}{2} - \frac{1}{2}] =: T_1$ if t is odd;
- $[u_i^{(1)}]_t \in \left[-\frac{t}{2} + 1, -\frac{t}{2} + 2, \ldots, \frac{t}{2}\right] =: T_2$ if t is even.

We have that $\#(T_1) = \#(T_2) = t$. Let $v_i^{(1)} := [u_i^{(1)}]_t$ for $0 \le i \le n-1$. It is clear that if $u_i^{(1)} = -t/2$, i.e. if $u_i^{(1)} = \lceil -t/2 \rceil$ and t is even, then $v_i^{(1)} = t/2$. We have

$$\begin{aligned} D(c(x) = 1) &= \left[u_0^{(1)} + u_1^{(1)}x + u_2^{(1)}x^2 + \cdots + u_{n-1}^{(1)}x^{n-1}\right]_t = [u_0^{(1)}]_t + [u_1^{(1)}]_t x + \cdots + [u_{n-1}^{(1)}]_t x^{n-1} \\ &= v_0^{(1)} + v_1^{(1)}x + v_2^{(1)}x^2 + \cdots + v_{n-1}^{(1)}x^{n-1} \end{aligned}$$

$$\text{where } \forall i = 0, 1, \ldots, n-1, \qquad v_i^{(1)} = \begin{cases} \frac{t}{2} & \text{if } u_i^{(1)} = -\frac{t}{2} (\text{i.e. if } u_i^{(1)} = \lceil -\frac{t}{2} \rceil \text{ and } t \text{ is even}) \\ u_i^{(1)} & \text{otherwise} \end{cases}$$

In particular, if t is odd, then $D(c = 1) = u_0^{(1)} + u_1^{(1)}x + u_2^{(1)}x^2 + \cdots + u_{n-1}^{(1)}x^{n-1}$.

We have, $\forall 0 \le i \le n-1$,

$$\text{if } t \text{ is odd, } -\frac{t}{2} + \frac{1}{2} \le v_i^{(1)} \le \frac{t}{2} - \frac{1}{2}; \text{ if } t \text{ is even, } -\frac{t}{2} + 1 \le v_i^{(1)} \le \frac{t}{2}$$

In both cases, $v_i^{(1)}$ can only have t different values. As we saw before, in case of t odd we need to perform $\lceil log_2(q/t)\rceil + 1$ oracle decryption queries; in case of t even, we need to perform extra oracle decryption queries (at most $n-1$) in order to understand which sign are given the coefficients of the secret key. Therefore, the total number of queries to the decryption oracle is at most $\lceil log_2(q/t)\rceil + n$. If we use the actual bound B given on the coefficients s_i by the distribution χ, we have that the total number of queries to the decryption oracle is at most $\lceil log_2(B/t)\rceil + n$.

4.2 Detailed Attack in three Cases

Case 1: t is odd

Step 1: select $c(x) = 1$ Select "ciphertext" $c(x) = 1$ and submit it to the decryption oracle. Since t is odd and $v_i^{(1)} = u_i^{(1)}$, $\forall 0 \le i \le n-1$, we obtain the polynomial $D(c = 1) = u_0^{(1)} + u_1^{(1)}x + u_2^{(1)}x^2 + \cdots + u_{n-1}^{(1)}x^{n-1}$, where $\lceil -\frac{t}{2}\rceil \le u_i^{(1)} \le \lfloor \frac{t}{2}\rfloor$. Every $u_i^{(1)}$ can have only t different values and can be written as $u_i^{(1)} = \lceil -\frac{t}{2}\rceil + k_{i,1}$, with $k_{i,1} \in \{0, 1, \ldots, t-1\}$. Now, it is easy to see that

$$u_i^{(1)} = \left\lceil -\frac{t}{2}\right\rceil + k_{i,1} \Leftrightarrow -\frac{q}{2} + \frac{q}{t}k_{i,1} < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,1}+1)$$

The polynomial obtained from the decryption oracle can therefore be written as

$$D(c(x) = 1) = u_0^{(1)} + u_1^{(1)}x + u_2^{(1)}x^2 + \cdots + u_{n-1}^{(1)}x^{n-1} = \sum_{i=0}^{n-1}\left(\left\lceil -\frac{t}{2}\right\rceil + k_{i,1}\right)x^i$$

Each f_i belongs to the interval $(-q/2, q/2)$. But after this our first query we learn values $k_{i,1} \in [0, 1, \ldots, t-1]$, $0 \le i \le n-1$, such that

$$-\frac{q}{2} + \frac{q}{t}k_{i,1} < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,1}+1) \qquad \text{(F(0,1))}$$

We have $-\frac{q}{2} + \frac{q}{t}(k_{i+1}+1) - \left(-\frac{q}{2} + \frac{q}{t}k_{i+1}\right) = \frac{q}{t}$. Therefore, we know each integer coefficient f_i with an error up to $\frac{q}{t}$. The idea now is to keep submitting 'ciphertext' to the decryption oracle and obtain values $k_{i,j}$, with $0 \le i \le n-1$ and increasing integers $j = 1, 2, 3, \ldots$, in such a way that we keep reducing the interval in which f_i lies until we know f_i with an error smaller than 1, which determines each f_i completely.

Step 2: select $c(x) = 2$ Select now "ciphertext" $c(x) = 2 = 2 + 0x + 0x^2 + \cdots + 0x^{n-1}$. Decryption oracle computes and return the polynomial

$$\begin{aligned} D(c=2) &= \left[\left\lfloor \frac{t}{q}\cdot [f\cdot 2]_q\right\rceil\right]_t = \left[\left\lfloor \frac{t}{q}\cdot\left([2f_0]_q + [2f_1]_q x + [2f_2]_q x^2 + \cdots + [2f_{n-1}]_q x^{n-1}\right)\right\rceil\right]_t \\ &= \left[\left\lfloor \frac{t}{q}f_0^{(2)}\right\rceil + \left\lfloor \frac{t}{q}f_1^{(2)}\right\rceil x + \cdots + \left\lfloor \frac{t}{q}f_{n-1}^{(2)}\right\rceil x^{n-1}\right]_t \end{aligned}$$

where we have put $f_i^{(2)} := [2f_i]_q$, for every $0 \le i \le n-1$; of course we have $-\frac{q}{2} < f_i^{(2)} < \frac{q}{2}$. Now,

- if $-q/4 < f_i < q/4$, then $-\frac{q}{2} < 2f_i < \frac{q}{2}$ and therefore $f_i^{(2)} = [2f_i]_q = 2f_i$
- if $-q/2 < f_i < -q/4$, then $-q < 2f_i < -\frac{q}{2}$ and therefore $f_i^{(2)} = [2f_i]_q = 2f_i + q$
- if $q/4 < f_i < q/2$, then $\frac{q}{2} < 2f_i < q$ and therefore $f_i^{(2)} = [2f_i]_q = 2f_i - q$

So we have

$$f_i^{(2)} = [2f_i]_q = \begin{cases} 2f_i & \text{if } -\frac{q}{4} < f_i < \frac{q}{4} \\ 2f_i + q & \text{if } -\frac{q}{2} < f_i < -\frac{q}{4}, \text{ and in this case } 0 < f_i^{(2)} < \frac{q}{2} \\ 2f_i - q & \text{if } \frac{q}{4} < f_i < \frac{q}{2}, \text{ and in this case } -\frac{q}{2} < f_i^{(2)} < 0 \end{cases} \qquad (2)$$

Let $u_i^{(2)} := \left\lfloor \frac{t}{q} \cdot f_i^{(2)} \right\rceil$. Then $D(c=2) = \left[u_0^{(2)} + u_1^{(2)} x + u_2^{(2)} x^2 + \cdots + u_{n-1}^{(2)} x^{n-1} \right]_t$. As before, $u_i^{(2)}$ can have only t different possible values, and can be written as $u_i^{(2)} = \left\lceil -\frac{t}{2} \right\rceil + k_{i,2}$, with $k_{i,2} \in \{0, 1, \ldots, t-1\}$, and also $u_i^{(2)} = \left\lceil -\frac{t}{2} \right\rceil + k_{i,2} \Leftrightarrow -\frac{q}{2} + \frac{q}{t} k_{i,2} < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,2} + 1)$. As before, since $-q/2 < f_i^{(2)} < q/2$ and t is odd, we have $\left\lceil -\frac{t}{2} \right\rceil \leq u_i^{(2)} \leq \left\lfloor \frac{t}{2} \right\rfloor$, and therefore we can simply write $D(c=2) = u_0^{(2)} + u_1^{(2)} x + u_2^{(2)} x^2 + \cdots + u_{n-1}^{(2)} x^{n-1} = \sum_{i=0}^{n-1} \left(\left\lceil -\frac{t}{2} \right\rceil + k_{i,2} \right) x^i$. So now, for each $0 \leq i \leq n-1$, we know $k_{i,1}, k_{i,2}$ such that

$$\begin{cases} -\frac{q}{2} + \frac{q}{t} k_{i,1} < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \\ -\frac{q}{2} + \frac{q}{t} k_{i,2} < [2f_i]_q < -\frac{q}{2} + \frac{q}{t}(k_{i,2} + 1) \end{cases}$$

There are 3 cases to distinguish, where $3 = 2^2 - 1$.

$(1/3)_{[c=2]}$. If $-\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \leq -\frac{q}{4} \wedge -\frac{q}{2} + \frac{q}{t} k_{i,1} \geq -\frac{q}{2}$, which says that $0 \leq k_{i,1} \leq \left\lfloor \frac{t}{4} - 1 \right\rfloor$, then we are sure that $f_i \in (-\frac{q}{2}, -\frac{q}{4})$. Therefore, by condition (2), we expect $f_i^{(2)} = [2f_i]_q = 2f_i + q$. Therefore, $-\frac{3q}{4} + \frac{q}{2t} k_{i,2} < f_i < -\frac{3q}{4} + \frac{q}{2t}(k_{i,2} + 1)$

$(2/3)_{[c=2]}$. If $-\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \leq \frac{q}{4} \wedge -\frac{q}{2} + \frac{q}{t} k_{i,1} \geq -\frac{q}{4}$, which says that $\left\lceil \frac{t}{4} \right\rceil \leq k_{i,1} \leq \left\lfloor \frac{3t}{4} - 1 \right\rfloor$, then we are sure that $f_i \in (-\frac{q}{4}, \frac{q}{4})$. Therefore, by condition (2), we expect $f_i^{(2)} = [2f_i]_q = 2f_i$. Therefore, $-\frac{q}{4} + \frac{q}{2t} k_{i,2} < f_i < -\frac{q}{4} + \frac{q}{2t}(k_{i,2} + 1)$

$(3/3)_{[c=2]}$. If $-\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \leq \frac{q}{2} \wedge -\frac{q}{2} + \frac{q}{t} k_{i,1} \geq \frac{q}{4}$, which says that $\left\lceil \frac{3t}{4} \right\rceil \leq k_{i,1} \leq t - 1$, then we are sure that $f_i \in (\frac{q}{4}, \frac{q}{2})$. Therefore, by condition (2), we expect $f_i^{(2)} = [2f_i]_q = 2f_i - q$. Therefore, $\frac{q}{4} + \frac{q}{2t} k_{i,2} < f_i < \frac{q}{4} + \frac{q}{2t}(k_{i,2} + 1)$

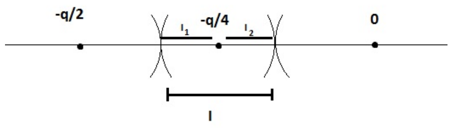
Now, we remark that there are values of $k_{i,1}$ for which is not clear to which of the previous cases we are falling in. For instance, if $k_{i,1}$ is such that $-\frac{q}{4} \in \left(-\frac{q}{2} + \frac{q}{t} k_{i,1}, -\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \right)$, then we are not sure whether we are in Case $(1/3)_{[c=2]}$ or in Case $(2/3)_{[c=2]}$. This uncertainty happens when $\nexists k_{i,1} \in [0, 1, \ldots, t-1]$ such that $-\frac{q}{2} + \frac{q}{t} k_{i,1} = -\frac{q}{4}$, i.e. such that $k_{i,1} = t/4$. So, if $\nexists k_{i,1} \in [0, 1, \ldots, t-1]$ such that $k_{i,1} = t/4$, i.e. if $4 \nmid t$, then $-\frac{q}{4} \in \left(-\frac{q}{2} + \frac{q}{t} \left\lfloor \frac{t}{4} \right\rfloor, -\frac{q}{2} + \frac{q}{t} \left(\left\lfloor \frac{t}{4} \right\rfloor + 1 \right) \right)$. So, if $k_{i,1} = \left\lfloor \frac{t}{4} \right\rfloor$, with $\frac{t}{4} \notin \mathbb{N}$, we have that $f_i \in \left(-\frac{q}{2} + \frac{q}{t} \left\lfloor \frac{t}{4} \right\rfloor, -\frac{q}{2} + \frac{q}{t} \left(\left\lfloor \frac{t}{4} \right\rfloor + 1 \right) \right) =: I$. It is easy to see that

$$-\frac{q}{2} + \frac{q}{t} \left(\left\lfloor \frac{t}{4} \right\rfloor + 1 \right) \leq 0, \forall 1 < t < q \tag{3}$$

There are two cases:

1/2: $f_i \in I_1 := I \cap (-q/2, -q/4)$. Then condition (2) implies that $f_i^{(2)} = [2f_i]_q \in (0, q/2)$

2/2: $f_i \in I_2 := I \cap (-q/4, 0)$. Then $f_i^{(2)} = [2f_i]_q \in (-q/2, 0)$



So, to sum up we have that if $k_{i,1} = \left\lfloor \frac{t}{4} \right\rfloor$, with $\frac{t}{4} \notin \mathbb{N}$, then

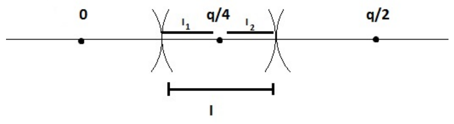
- if $f_i^{(2)} \in (0, q/2)$ then $f_i \in (-q/2, -q/4)$ and apply Case $(1/3)_{[c=2]}$
- if $f_i^{(2)} \in (-q/2, 0)$ then $f_i \in (-q/4, 0)$ and apply Case $(2/3)_{[c=2]}$

Similarly to what just discussed, if $k_{i,1}$ is such that $\frac{q}{4} \in \left(-\frac{q}{2} + \frac{q}{t} k_{i,1}, -\frac{q}{2} + \frac{q}{t}(k_{i,1} + 1) \right)$, then we are not sure if we are in Case $(2/3)_{[c=2]}$ or in Case $(3/3)_{[c=2]}$ This uncertainty happens when $\nexists k_{i,1} \in [0, 1, \ldots, t-1]$ such that $-\frac{q}{2} + \frac{q}{t} k_{i,1} = \frac{q}{4}$, i.e. such that $k_{i,1} = 3t/4$. So, if $\nexists k_{i,1} \in [0, 1, \ldots, t-1]$ such that $k_{i,1} = 3t/4$, then $\frac{q}{4} \in \left(-\frac{q}{2} + \frac{q}{t} \left\lfloor \frac{3t}{4} \right\rfloor, -\frac{q}{2} + \frac{q}{t} \left(\left\lfloor \frac{3t}{4} \right\rfloor + 1 \right) \right)$. So, if $k_{i,1} = \left\lfloor \frac{3t}{4} \right\rfloor$, with $\frac{3t}{4} \notin \mathbb{N}$, we have that $f_i \in \left(-\frac{q}{2} + \frac{q}{t} \left\lfloor \frac{3t}{4} \right\rfloor, -\frac{q}{2} + \frac{q}{t} \left(\left\lfloor \frac{3t}{4} \right\rfloor + 1 \right) \right) =: I$. It is easy to see that

$$-\frac{q}{2} + \frac{q}{t} \left\lfloor \frac{3t}{4} \right\rfloor \geq 0, \forall t, q \tag{4}$$

There are two cases:

1/2: $f_i \in I_1 := I \cap (0, q/4)$. Then $f_i^{(2)} = [2f_i]_q \in (0, q/2)$
2/2: $f_i \in I_2 := I \cap (q/4, q/2)$. Then condition (2) implies that $f_i^{(2)} = [2f_i]_q \in (-q/2, 0)$



So, to sum up we have that if $k_{i,1} = \lfloor \frac{3t}{4} \rfloor$, with $\frac{3t}{4} \notin \mathbb{N}$, then

- if $f_i^{(2)} \in (0, q/2)$ then $f_i \in (-q/4, q/4)$ and apply Case $(2/3)_{[c=2]}$
- if $f_i^{(2)} \in (-q/2, 0)$ then $f_i \in (q/4, q/2)$ and apply Case $(3/3)_{[c=2]}$

We can write now all the 3 cases in a more complete way:

$(1/3)_{[c=2]}$. Suppose that

$$0 \le k_{i,1} \le \left\lfloor \frac{t}{4} - 1 \right\rfloor \vee \left(k_{i,1} = \left\lfloor \frac{t}{4} \right\rfloor, \text{with } \frac{t}{4} \notin \mathbb{N} \wedge f_i^{(2)} \in (0, q/2) \right) \tag{K(1,1)}$$

Then

$$f_i \in \left(-\frac{q}{2}, -\frac{q}{4}\right), \qquad -\frac{3q}{4} + \frac{q}{2t} k_{i,2} < f_i < -\frac{3q}{4} + \frac{q}{2t}(k_{i,2}+1) \tag{F((1,1))}$$

$(2/3)_{[c=2]}$. Suppose that

$$\left\lceil \frac{t}{4} \right\rceil \le k_{i,1} \le \left\lfloor \frac{3t}{4} - 1 \right\rfloor \vee \left(k_{i,1} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(2)} \in (-q/2, 0) \right) \vee \\ \vee \left(k_{i,1} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(2)} \in (0, q/2) \right) \tag{K(1,2)}$$

Then

$$f_i \in \left(-\frac{q}{4}, \frac{q}{4}\right), \qquad -\frac{q}{4} + \frac{q}{2t} k_{i,2} < f_i < -\frac{q}{4} + \frac{q}{2t}(k_{i,2}+1) \tag{F(1,2)}$$

$(3/3)_{[c=2]}$. Suppose that

$$\left\lceil \frac{3t}{4} \right\rceil \le k_{i,1} \le t-1 \vee \left(k_{i,1} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(2)} \in (-q/2, 0) \right) \tag{K(1,3)}$$

Then

$$f_i \in \left(\frac{q}{4}, \frac{q}{2}\right), \qquad \frac{q}{4} + \frac{q}{2t} k_{i,2} < f_i < \frac{q}{4} + \frac{q}{2t}(k_{i,2}+1) \tag{F(1,3)}$$

In all cases, we end up by knowing f_i with an error up to $\frac{q}{2t}$.

Step 3: select $c(x) = 4$ Select now "ciphertext" $c(x) = 4 = 4 + 0x + 0x^2 + \cdots + 0x^{n-1}$. Decryption oracle computes and return the polynomial

$$\begin{aligned} D(c=4) &= \left[\left\lfloor \frac{t}{q} \cdot [f \cdot 4]_q \right\rceil \right]_t = \left[\left\lfloor \frac{t}{q} \cdot \left([4f_0]_q + [4f_1]_q x + [4f_2]_q x^2 + \cdots + [4f_{n-1}]_q x^{n-1} \right) \right\rceil \right]_t \\ &= \left[\left\lfloor \frac{t}{q} f_0^{(3)} \right\rceil + \left\lfloor \frac{t}{q} f_1^{(3)} \right\rceil x + \cdots + \left\lfloor \frac{t}{q} f_{n-1}^{(3)} \right\rceil x^{n-1} \right]_t \end{aligned}$$

where we have put $f_i^{(3)} := [4f_i]_q$, for every $0 \le i \le n-1$; of course we have $-\frac{q}{2} < f_i^{(3)} < \frac{q}{2}$. In a similar way to what we have seen before, we have that

$$D(c=4) = u_0^{(3)} + u_1^{(3)} x + u_2^{(3)} x^2 + \cdots + u_{n-1}^{(3)} x^{n-1} = \sum_{i=0}^{n-1} \left(\left\lceil -\frac{t}{2} \right\rceil + k_{i,3} \right)$$

and therefore we learn integers $k_{i,3}$, for $0 \le i \le n-1$. Now, for each $0 \le i \le n-1$, we know $k_{i,1}, k_{i,2}, k_{i,3}$ such that

$$\begin{cases} -\frac{q}{2} + \frac{q}{t}k_{i,1} < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,1}+1) \\ -\frac{q}{2} + \frac{q}{t}k_{i,2} < [2f_i]_q < -\frac{q}{2} + \frac{q}{t}(k_{i,2}+1) \\ -\frac{q}{2} + \frac{q}{t}k_{i,3} < [4f_i]_q < -\frac{q}{2} + \frac{q}{t}(k_{i,3}+1) \end{cases}$$

There are $7 = 2^3 - 1$ cases to distinguish. As before, we obtain

$(1/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,1)) holds}] \wedge \left(\left\lceil \frac{t}{2} \right\rceil \le k_{i,2} \le \left\lfloor \frac{3t}{4} - 1 \right\rfloor \vee \left(k_{i,2} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(0, \frac{q}{2}\right)\right)\right) \tag{K(2,1)}$$

Then

$$f_i \in (-\frac{q}{2}, -\frac{3q}{8}), \qquad -\frac{5q}{8} + \frac{q}{4t}k_{i,3} < f_i < -\frac{5q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,1)}$$

$(2/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,1)) holds}] \wedge \left(\left\lceil \frac{3t}{4} \right\rceil \le k_{i,2} \le t-1 \vee \left(k_{i,2} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(-\frac{q}{2}, 0\right)\right)\right) \tag{K(2,2)}$$

Then

$$f_i \in (-\frac{3q}{8}, -\frac{q}{4}), \qquad -\frac{3q}{8} + \frac{q}{4t}k_{i,3} < f_i < -\frac{3q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,2)}$$

$(3/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,2)) holds}] \wedge \left(0 \le k_{i,2} \le \left\lfloor \frac{t}{4} - 1 \right\rfloor \vee \left(k_{i,2} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(0, \frac{q}{2}\right)\right)\right) \tag{K(2,3)}$$

Then

$$f_i \in (-\frac{q}{4}, -\frac{q}{8}), \qquad -\frac{3q}{8} + \frac{q}{4t}k_{i,3} < f_i < -\frac{3q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,3)}$$

$(4/7)_{[c=4]}$. Suppose that

$$\begin{aligned} [\text{Condition (K(1,2)) holds}] \wedge \Big[\left\lceil \frac{t}{4} \right\rceil \le k_{i,2} \le \left\lfloor \frac{3t}{4} - 1 \right\rfloor \vee \left(k_{i,2} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(-\frac{q}{2}, 0\right)\right) \vee \\ \vee \left(k_{i,2} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(0, \frac{q}{2}\right)\right) \Big] \end{aligned} \tag{K(2,4)}$$

Then

$$f_i \in (-\frac{q}{8}, \frac{q}{8}), \qquad -\frac{q}{8} + \frac{q}{4t}k_{i,3} < f_i < -\frac{q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,4)}$$

$(5/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,2)) holds}] \wedge \left(\left\lceil \frac{3t}{4} \right\rceil \le k_{i,2} \le t-1 \vee \left(k_{i,2} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(-\frac{q}{2}, 0\right)\right)\right) \tag{K(2,5)}$$

Then

$$f_i \in (\frac{q}{8}, \frac{q}{4}), \qquad \frac{q}{8} + \frac{q}{4t}k_{i,3} < f_i < \frac{q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,5)}$$

$(6/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,3)) holds}] \wedge \left(0 \le k_{i,2} \le \left\lfloor \frac{t}{4} - 1 \right\rfloor \vee \left(k_{i,2} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(0, \frac{q}{2}\right)\right)\right) \tag{K(2,6)}$$

Then

$$f_i \in (\frac{q}{4}, \frac{3q}{8}), \qquad \frac{q}{8} + \frac{q}{4t}k_{i,3} < f_i < \frac{q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,6)}$$

$(7/7)_{[c=4]}$. Suppose that

$$[\text{Condition (K(1,3)) holds}] \ \wedge \left(\left\lceil \frac{t}{4} \right\rceil \leq k_{i,2} \leq \left\lfloor \frac{t}{2} - 1 \right\rfloor \vee \left(k_{i,2} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(3)} \in \left(-\frac{q}{2}, 0 \right) \right) \right) \tag{K(2,7)}$$

Then

$$f_i \in (\frac{3q}{8}, \frac{q}{2}), \qquad \frac{3q}{8} + \frac{q}{4t} k_{i,3} < f_i < \frac{3q}{8} + \frac{q}{4t}(k_{i,3}+1) \tag{F(2,7)}$$

In all cases, we end up by knowing f_i with an error up to $\frac{q}{4t}$.

Generalization and complexity At each step, we keep submitting "ciphertexts" $c(x) := 2^h$, for increasing values $h = 0, 1, 2, \ldots$, i.e. at step $h+1$ we submit ciphertext $c(x) = 2^h$. Suppose we are at step $h+1$. Then we submit to the decryption oracle the 'ciphertext' $c(x) = 2^h$, and the decryption oracle will return us a polynomial

$$D(c = 2^h) = u_0^{(h+1)} + u_1^{(h+1)} x + \cdots + u_{n-1}^{(h+1)} x^{n-1} = \sum_{i=0}^{n-1} u_i^{(h+1)} x^i = \sum_{i=0}^{n-1} \left(\left\lceil -\frac{t}{2} \right\rceil + k_{i,h+1} \right) \in R_t$$

from which we learn values $k_{i,h+1}$ for $1 \leq i \leq n-1$. So, at this point, we know $k_{i,j}$, for $0 \leq i \leq n-1$ and $1 \leq j \leq h+1$. These values allow us to distinguish between $m_h := 2^{h+1} - 1$ cases: for each $0 \leq i \leq n-1$, we know that integer f_i belongs to one of the cases:

$(a/2^{h+1} - 1)_{[c=2^h]}$. Suppose that

$$[\text{Condition (C}(h, a, 1)\text{) holds}] \ \wedge [\text{Condition (C}(h, a, 2)\text{) holds}] \tag{K(h,a)}$$

Then

$$f_i \in (x_{a,h}, y_{a,h}), \qquad \Delta_{h,a} + \frac{q}{2^h t} k_{i,h+1} < f_i < \Delta_{h,a} + \frac{q}{2^h t}(k_{i,h+1} + 1) \tag{F(h,a)}$$

where $a \in \{1, 2, \ldots, 2^{h+1} - 1\}$. Since

$$\Delta_{h,a} + \frac{q}{2^h t}(k_{i,h+1} + 1) - \left(\Delta_{h,a} + \frac{q}{2^h t} k_{i,h+1} \right) = \frac{q}{2^h t},$$

this allows us to recover, for each $0 \leq i \leq n-1$, the integer f_i with an error up to $\frac{q}{2^h t}$. Therefore, we keep submitting 'ciphertexts' $c(x) = 2^h$ for increasing values $h = 0, 1, 2, \ldots$ until h is such that $\frac{q}{2^h t} < 1$, i.e. $h \geq \lceil \log_2(q/t) \rceil$. So, we have to repeat our attack, submitting ciphertexts $c(x) = 1 = 2^0, 2^1, 2^2, 2^3, \ldots, 2^H$, where $H := \lceil \log_2(q/t) \rceil$. Se we repeat our attack $H+1$ times. Now, the secret key is $f(x) = f_0 + f_1 x + \cdots + f_{n-1} x^{n-1}$, where $f_i \in (-q/2, q/2]$, $\forall 0 \leq i \leq n-1$. So f_i can have q different values. The decryption oracle reveals a polynomial $m(x) = m_0 + m_1 x + \cdots + m_{n-1} x^{n-1}$, where $m_i \in (-t/2, t/2]$, $\forall 0 \leq i \leq n-1$. So m_i can have t different values. Each f_i can be described with at most $\lfloor \log_2(q-1) \rfloor + 1$ bits. So $f(x)$ can be described with $n \cdot (\lfloor \log_2(q-1) \rfloor + 1)$. Oracle decryption reveals $n \cdot (\lfloor \log_2(t-1) \rfloor + 1)$ bits. So the minimum number of oracle queries to determine $f(x)$ is given by $\frac{n \cdot (\lfloor \log_2(q-1) \rfloor + 1)}{n \cdot (\lfloor \log_2(t-1) \rfloor + 1)}$. In order to finish our attack for t odd, we need to give complete description of $\Delta_{h,a}$, Condition C$(h, a, 1)$ and Condition C$(h, a, 2)$, for each $0 \leq h \leq \lceil \log_2(q/t) \rceil = H$ and for each $1 \leq a \leq 2^{h+1} - 1$. Fix $0 \leq h \leq \lceil \log_2(q/t) \rceil$. For a given $1 \leq a \leq 2^{h+1} - 1$ put

$$\delta_{h,a} := \begin{cases} 2^{h-1} & \text{if } a = 2^h \\ \lfloor \frac{a}{2} \rfloor & \text{if } 1 \leq a < 2^h \\ \lceil \frac{a}{2} \rceil & \text{if } 2^h < a \leq 2^{h+1} - 1 \end{cases}, \quad \Delta_{h,a} := -\left(\frac{1}{2} + \frac{1}{2^{h+1}} - \frac{\delta_{h,a}}{2^h} \right) \cdot q$$

Also, put

$$\eta(h, a) := \begin{cases} \lceil \frac{a}{2} \rceil & \text{if } 1 \leq a \leq 2^h \\ \lfloor \frac{a}{2} \rfloor & \text{if } 2^h < a \leq 2^{h+1} - 1 \end{cases}$$

Then

$$\text{Condition (C}(h, a, 1)) = \text{Condition (K}(h-1, \eta(h, a)))$$

Remark that, if $h = 0$ or $h = 1$, then Condition $(\mathrm{C}(h, a, 1)) = \emptyset$ i.e., we don't put any condition at all, vacuous condition.

For Condition $\mathrm{C}(h, a, 2)$, remark that if $h = 0$ then Condition $(\mathrm{C}(0, a, 2)) = \emptyset$ i.e., we don't put any condition at all, vacuous condition. One can see that, at step $h+1$, condition $\mathrm{C}(h, a, 2)$ is only one among the following 5:

1. $V_{3,h} := U_{2,1} = U_{1,1} \wedge (r \text{ is even}) = U_{3,1} \wedge (r \text{ is odd})$:
$$0 \leq k_{i,h} \leq \left\lfloor \frac{t}{4} - 1 \right\rfloor \vee \left(k_{i,h} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(0, \frac{q}{2}\right) \right) \tag{$V_{3,h}$}$$

2. $V_{5,h} := U_{2,2}$:
$$\left\lceil \frac{t}{4} \right\rceil \leq k_{i,h} \leq \left\lfloor \frac{3t}{4} - 1 \right\rfloor \vee \left(k_{i,h} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(-\frac{q}{2}, 0\right) \right) \vee \\ \vee \left(k_{i,h} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(0, \frac{q}{2}\right) \right) \tag{$V_{5,h}$}$$

3. $V_{2,h} := U_{2,3} = U_{1,2} \wedge (r \text{ is odd}) = U_{3,2} \wedge (r \text{ is even})$:
$$\left\lceil \frac{3t}{4} \right\rceil \leq k_{i,h} \leq t - 1 \vee \left(k_{i,h} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(-\frac{q}{2}, 0\right) \right) \tag{$V_{2,h}$}$$

4. $V_{1,h} := U_{1,1} \wedge (r \text{ is odd}) = U_{3,1} \wedge (r \text{ is even})$:
$$\left\lceil \frac{t}{2} \right\rceil \leq k_{i,h} \leq \left\lfloor \frac{3t}{4} - 1 \right\rfloor \vee \left(k_{i,h} = \left\lfloor \frac{3t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(0, \frac{q}{2}\right) \right) \tag{$V_{1,h}$}$$

5. $V_{0,h} := U_{1,2} \wedge (r \text{ is even}) = U_{3,2} \wedge (r \text{ is odd})$:
$$\left\lceil \frac{t}{4} \right\rceil \leq k_{i,h} \leq \left\lfloor \frac{t}{2} - 1 \right\rfloor \vee \left(k_{i,h} = \left\lfloor \frac{t}{4} \right\rfloor \wedge f_i^{(h+1)} \in \left(-\frac{q}{2}, 0\right) \right) \tag{$V_{0,h}$}$$

So, suppose we are in case $(a/2^{h+1} - 1)_{[c=2^h]}$. Then we see that we have

Therefore, we have

$$\mathrm{C}(h, a, 2) = \begin{cases} V_{1,h} & \text{if } 1 \leq a \leq 2^h - 2 \wedge a \equiv 1 \bmod 4 \textbf{ or } 2^h + 2 \leq a \leq 2^{h+1} - 1 \wedge a \equiv 0 \bmod 4 \\ V_{2,h} & \text{if } 1 \leq a \leq 2^h - 2 \wedge a \equiv 2 \bmod 4 \textbf{ or } 2^h + 2 \leq a \leq 2^{h+1} - 1 \wedge a \equiv 1 \bmod 4 \\ & \qquad \textbf{or } a = 2^h + 1 \\ V_{3,h} & \text{if } 1 \leq a \leq 2^h - 2 \wedge a \equiv 3 \bmod 4 \textbf{ or } 2^h + 2 \leq a \leq 2^{h+1} - 1 \wedge a \equiv 2 \bmod 4 \\ & \qquad \textbf{or } a = 2^h - 1 \\ V_{0,h} & \text{if } 1 \leq a \leq 2^h - 2 \wedge a \equiv 0 \bmod 4 \textbf{ or } 2^h + 2 \leq a \leq 2^{h+1} - 1 \wedge a \equiv 3 \bmod 4 \\ V_{5,h} & \text{if } a = 2^h \end{cases}$$

Case 2: t is even but not 2

Step 1: select $c(x) = 1$ Select "ciphertext" $c(x) = 1$ and submit it to the decryption oracle. We obtain the polynomial $D(c(x) = 1) = v_0^{(1)} + v_1^{(1)} x + v_2^{(1)} x^2 + \cdots + v_{n-1}^{(1)} x^{n-1}$. Suppose there exists $v_i^{(1)} = t/2$. This means that either $u_i^{(1)} = \frac{t}{2}$ or $u_i^{(1)} = -\frac{t}{2}$. We want to find out which one among the two above cases holds.

1. If we are in case $u_i^{(1)} = \frac{t}{2}$, then we have $\left\lfloor \frac{t}{q} f_i \right\rceil = \frac{t}{2} \Leftrightarrow \frac{q}{2} - \frac{q}{2t} < f_i < \frac{q}{2}$
2. If we are in case $u_i^{(1)} = -\frac{t}{2}$, then we have $\left\lfloor \frac{t}{q} f_i \right\rceil = -\frac{t}{2} \Leftrightarrow -\frac{q}{2} < f_i < -\frac{q}{2} + \frac{q}{2t}$

To find out which one is the case, we have to wait for the next step.

Now, let's focus on all the other $v_i^{(1)} \neq \frac{t}{2}$. We have in this case, $v_i^{(1)} = u_i^{(1)}$. Now, similarly as before, we have $-\frac{t}{2}+1 \leq u_i^{(1)} \leq \frac{t}{2}$, and every $u_i^{(1)}$ can have only t different values; it can be written as $u_i^{(1)} = -\frac{t}{2}+1+k_{i,1}, \quad$ with $k_{i,1} \in \{0,1,\dots,t-1\}$. Now, it is easy to see that

$$u_i^{(1)} = -\frac{t}{2}+1+k_{i,1} \Leftrightarrow -\frac{q}{2}+\frac{q}{t}(k_{i,1}+\frac{1}{2}) < f_i < -\frac{q}{2}+\frac{q}{t}(k_{i,1}+\frac{3}{2})$$

The polynomial obtained from the decryption oracle can therefore be written as $D(c(x)=1) = \sum_{i=0}^{n-1}\left(-\frac{t}{2}+1+k_{i,1}\right)x^i$. Each f_i belongs to the interval $(-q/2, q/2)$. But after this our first query we learn values $k_{i,1} \in [0,1,\dots,t-1]$, $0 \leq i \leq n-1$, such that $-\frac{q}{2}+\frac{q}{t}(k_{i,1}+\frac{1}{2}) < f_i < -\frac{q}{2}+\frac{q}{t}(k_{i,1}+\frac{3}{2})$ We have that $-\frac{q}{2}+\frac{q}{t}(k_{i+1}+3/2) - \left(-\frac{q}{2}+\frac{q}{t}(k_{i+1}+1/2)\right) = \frac{q}{t}$. Therefore, we know each integer coefficient f_i with an error up to $\frac{q}{t}$.

The idea now is to keep submitting 'ciphertext' to the decryption oracle and obtain values $k_{i,j}$, with $0 \leq i \leq n-1$ and increasing integers $j = 1,2,3,\dots$, in such a way that we keep reducing the interval in which f_i lies until we know f_i with an error smaller than 1, which determines each f_i completely.

Step 2: select $c(x) = 2$ Select now "ciphertext" $c(x) = 2 = 2+0x+\cdots+0x^{n-1}$. Decryption oracle computes and return the polynomial

$$D(c(x)=2) = \left[\left\lfloor\frac{t}{q}\cdot[f\cdot 2]_q\right\rceil\right]_t = \left[\left\lfloor\frac{t}{q}\cdot\left([2f_0]_q+[2f_1]_q x+\cdots+[2f_{n-1}]_q x^{n-1}\right)\right\rceil\right]_t$$

Now, let's focus on $\left[\left\lfloor\frac{t}{q}[2f_i]_q\right\rceil\right]_t x^i$ for each i such that, in the previous step, $v_i^{(1)} = \frac{t}{2}$.

1. We have

$$\begin{aligned}\frac{q}{2}-\frac{q}{2t} < f_i < \frac{q}{2} &\Leftrightarrow q-\frac{q}{t} < 2f_i < q \Leftrightarrow -\frac{q}{t} < [2f_i]_q < 0\\ &\Leftrightarrow -1 < \frac{t}{q}[2f_i]_q < 0 \Leftrightarrow -1 \leq \left[\left\lfloor\frac{t}{q}[2f_i]_q\right\rceil\right]_t \leq 0\\ &\Leftrightarrow \left[\left\lfloor\frac{t}{q}[2f_i]_q\right\rceil\right]_t = \begin{cases}0 \text{ or } -1 & \text{if } t>2\\ 0 \text{ or } 1 & \text{if } t=2\end{cases}\end{aligned}$$

2. We have analogously $-\frac{q}{2} < f_i < -\frac{q}{2}+\frac{q}{2t} \Leftrightarrow \left[\left\lfloor\frac{t}{q}[2f_i]_q\right\rceil\right]_t = 0 \text{ or } 1$.

From now on we assume $t > 2$; we will consider later the case in which $t = 2$. Let $v_i^{(2)} = \left[\left\lfloor\frac{t}{q}[2f_i]_q\right\rceil\right]_t$. We have that

1. if $v_i^{(2)} = -1$, then $u_i^{(1)} = \frac{t}{2}$ and $\frac{q}{2}-\frac{q}{2t} < f_i < \frac{q}{2}$
2. if $v_i^{(2)} = 1$, then $u_i^{(1)} = -\frac{t}{2}$ and $-\frac{q}{2} < f_i < -\frac{q}{2}+\frac{q}{2t}$
3. if $v_i^{(2)} = 0$, then we can't conclude right now the exact interval in which f_i belongs; this will be considered in the next step.

Remark 2. Suppose we are in the above case 3, i.e. $v^{(2)} = \left\lfloor\frac{t}{q}[2f_i]_q\right\rceil = 0$. Then

1. We have

$$\frac{q}{2}-\frac{q}{2t} < f_i < \frac{q}{2} \quad \wedge \quad \left\lfloor\frac{t}{q}[2f_i]_q\right\rceil = 0 \Leftrightarrow \frac{q}{2}-\frac{q}{4t} < f_i < \frac{q}{2}$$

2. Similarly, we have

$$-\frac{q}{2} < f_i < -\frac{q}{2}+\frac{q}{2t} \quad \wedge \quad \left\lfloor\frac{t}{q}[2f_i]_q\right\rceil = 0 \Leftrightarrow -\frac{q}{2} < f_i < -\frac{q}{2}+\frac{q}{4t}$$

We will use this remark in the next step to investigate further the interval in which f_i lies. Now, let's focus on all of the other coefficients. Using the same arguments as in section 4.2, the decryption oracle computes and return the polynomial

$$D(c(x)=2) = \left[\left\lfloor \frac{t}{q} \cdot [f \cdot 2]_q \right\rceil \right]_t = \left[\left\lfloor \frac{t}{q} f_0^{(2)} \right\rceil + \left\lfloor \frac{t}{q} f_1^{(2)} \right\rceil x + \cdots + \left\lfloor \frac{t}{q} f_{n-1}^{(2)} \right\rceil x^{n-1} \right]_t$$
$$= \left[u_0^{(2)} + u_1^{(2)} x + u_2^{(2)} x^2 + \cdots + u_{n-1}^{(2)} x^{n-1} \right]_t := v_0^{(2)} + v_1^{(2)} x + \cdots + v_{n-1}^{(2)} x^{n-1}$$

As before, suppose there exists $v_i^{(2)} = t/2$. This means that either $u_i^{(2)} = \frac{t}{2}$, or $u_i^{(2)} = -\frac{t}{2}$. We can easily understand which case we are by considering the known value $v_i^{(1)} \neq \frac{t}{2}$. All the other $v_i^{(2)}$ correspond to values $u_i^{(2)} \neq \frac{-t}{2}$. These $u_i^{(2)}$ can then have only t different possible values, and can be written as $u_i^{(2)} = -\frac{t}{2} + 1 + k_{i,2}, \quad$ with $k_{i,2} \in \{0, 1, \ldots, t-1\}$, and also

$$u_i^{(2)} = -\frac{t}{2} + 1 + k_{i,2} \Leftrightarrow -\frac{q}{2} + \frac{q}{t}(k_{i,2} + \frac{1}{2}) < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,2} + \frac{3}{2})$$

So now, for each $0 \leq i \leq n-1$ such that $v_i^{(1)} \neq \frac{t}{2} \vee (v_i^{(1)} = \frac{t}{2} \wedge v_i^{(2)} = 0)$, we know $k_{i,1}, k_{i,2}$ such that

$$\begin{cases} -\frac{q}{2} + \frac{q}{t}(k_{i,1} + \frac{1}{2}) < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,1} + \frac{3}{2}) \\ -\frac{q}{2} + \frac{q}{t}(k_{i,2} + \frac{1}{2}) < f_i < -\frac{q}{2} + \frac{q}{t}(k_{i,2} + \frac{3}{2}) \end{cases}$$

There are 3 cases to distinguish. These cases can be computed in an analogous way to what seen for the case t odd. We omit the details.

Generalization We continue in this way, following the blueprint for t odd and taking care of all the coefficients for which $v_i^{(1)} = \frac{t}{2}$ and all subsequents $v_i^{(j)} = 0$ (when we finally find a $j \geq 2$ such that $v_i^{(j)} = 1$ or -1, then we can deduce the original value of $u_i^{(1)} = \frac{t}{2}$ or $-\frac{t}{2}$). If at the last step m we still get $v_i^{(m)} = 0$, then all the values $u_i^{(1)}$ remain undetermined, which also say that all the corresponding coefficients f_i can have only two possible values. At this point, the strategy is to submit to the decryption oracle 'ciphertexts' in order to determine whether $f_i \cdot f_j < 0$ or $f_i \cdot f_j > 0$ holds among all the non-zero coefficients f_i, f_j, in a way similar to what we have already discussed for the attack on the [LATV12] SHW scheme. We omit the details; we will give a description of how to do this in the case $t = 2$; the general case $t > 2$ is then easy to obtain. We study now the case $t = 2$.

Case 3: $\mathbf{t = 2}$

Step 1: select $\boldsymbol{c(x) = 1}$ Choose and submit to the decryption oracle the polynomial $c(x) = 1$. It will compute and return the polynomial

$$D(c(x)=1) = \left[\left\lfloor \frac{2}{q} \cdot [f \cdot 1]_q \right\rceil \right]_2 = \left[\left\lfloor \frac{2}{q} f_0 \right\rceil + \left\lfloor \frac{2}{q} f_1 \right\rceil x + \cdots + \left\lfloor \frac{2}{q} f_{n-1} \right\rceil x^{n-1} \right]_2$$

For every $0 \leq i \leq n-1$, $u_i^{(1)} := \left\lfloor \frac{2}{q} f_i \right\rceil$ is such that $-1 \leq u_i^{(1)} \leq 1$, and so $v_i^{(1)} := [u_i^{(1)}]_2 = 0$ or 1. We have two cases to distinguish:

1) $v_i^{(1)} = 0$. We have $v_i^{(1)} = 0 \Leftrightarrow u_i^{(1)} = 0 \Leftrightarrow \left\lfloor \frac{2}{q} f_i \right\rceil = 0 \Leftrightarrow -\frac{1}{2} < \frac{2}{q} f_i < \frac{1}{2} \Leftrightarrow -\frac{q}{4} < f_i < \frac{q}{4}$

2) $v_i^{(1)} = 1$. We have

$$v_i^{(1)} = 1 \Leftrightarrow u_i^{(1)} = -1 \text{ or } u_i^{(1)} = +1 \Leftrightarrow \left\lfloor \frac{2}{q} f_i \right\rceil = -1 \text{ or } \left\lfloor \frac{2}{q} f_i \right\rceil = +1$$
$$\Leftrightarrow -\frac{3}{2} < \frac{2}{q} f_i < -\frac{1}{2} \text{ or } \frac{1}{2} < \frac{2}{q} f_i < \frac{3}{2} \Leftrightarrow -\frac{q}{2} < f_i < -\frac{q}{4} \text{ or } \frac{q}{4} < f_i < \frac{q}{2}$$

Step 2: select $c(x) = 2$ Choose and submit to the decryption oracle the polynomial $c(x) = 2$. It will compute and return the polynomial $D(c(x) = 2) = \sum_{i=0}^{n-1} \left[\left\lfloor \frac{2}{q}[2f_i]_q \right\rceil\right]_2 x^i =: \sum_{i=0}^{n-1} \left[u_i^{(2)}\right]_2 x^i =: \sum_{i=0}^{n-1} v_i^{(2)} x^i$. We have two cases to distinguish:

1) $v_i^{(2)} = 0$. We have

$$v_i^{(2)} = 0 \Leftrightarrow u_i^{(2)} = 0 \Leftrightarrow \left\lfloor \frac{2}{q}[2f_i]_q \right\rceil = 0 \Leftrightarrow -\frac{1}{2} < \frac{2}{q}[2f_i]_q < \frac{1}{2} \Leftrightarrow -\frac{q}{4} < [2f_i]_q < \frac{q}{4}$$
$$\Leftrightarrow -\frac{q}{4} < 2f_i < \frac{q}{4} \text{ or } -\frac{5q}{4} < 2f_i < -\frac{3q}{4} \text{ or } \frac{3q}{4} < 2f_i < \frac{5q}{4}$$
$$\Leftrightarrow -\frac{q}{8} < f_i < \frac{q}{8} \text{ or } -\frac{q}{2} < f_i < -\frac{3q}{8} \text{ or } \frac{3q}{8} < f_i < \frac{q}{2}$$

We have three cases to distinguish, according to which known interval f_i lies at the end of step 1:

1.1) If $-\frac{q}{4} < f_i < \frac{q}{4}$, then $-\frac{q}{8} < f_i < \frac{q}{8}$
1.2) If $-\frac{q}{2} < f_i < -\frac{q}{4}$, then $-\frac{q}{2} < f_i < -\frac{3q}{8}$
1.3)] If $\frac{q}{4} < f_i < \frac{q}{2}$, then $\frac{3q}{8} < f_i < \frac{q}{2}$

2) $v_i^{(2)} = 1$. We have

$$v_i^{(2)} = 1 \Leftrightarrow u_i^{(2)} = -1 \text{ or } u_i^{(2)} = +1 \Leftrightarrow \left\lfloor \frac{2}{q}[2f_i]_q \right\rceil = -1 \text{ or } \left\lfloor \frac{2}{q}[2f_i]_q \right\rceil = +1$$
$$\Leftrightarrow -\frac{3}{2} < \frac{2}{q}[2f_i]_q < -\frac{1}{2} \text{ or } \frac{1}{2} < \frac{2}{q}[2f_i]_q < \frac{3}{2}$$
$$\Leftrightarrow -\frac{3q}{4} < [2f_i]_q < -\frac{q}{4} \text{ or } \frac{q}{4} < [2f_i]_q < \frac{3q}{4}$$
$$\Leftrightarrow -\frac{3q}{4} < 2f_i < -\frac{q}{4} \text{ or } \frac{q}{4} < 2f_i < \frac{3q}{4}$$
$$\Leftrightarrow -\frac{3q}{8} < f_i < -\frac{q}{8} \text{ or } \frac{q}{8} < f_i < \frac{3q}{8}$$

Now, again we have three cases to distinguish, according to which known interval f_i lies at the end of step 1:

2.1) If $-\frac{q}{4} < f_i < \frac{q}{4}$, then $-\frac{q}{4} < f_i < -\frac{q}{8}$ or $\frac{q}{8} < f_i < \frac{q}{4}$
2.2) If $-\frac{q}{2} < f_i < -\frac{q}{4}$, then $-\frac{3q}{8} < f_i < -\frac{q}{4}$
2.3)] If $\frac{q}{4} < f_i < \frac{q}{2}$, then $\frac{q}{4} < f_i < \frac{3q}{8}$

Generalization and the last step We continue in this way, and in the end we will know each coefficient f_i up to the sign. Therefore, we will know a polynomial $f'(x) = f'_0 + f'_1 x + \cdots + f'_{n-1}x^{n-1}$, with $f'_i = |f_i|$ for every i. We proceed similarly to what we have seen for the attack on the [LATV12] scheme, i.e. we query the decryption oracle in order to find out the relations $f_i \cdot f_j < 0$ or $f_i \cdot f_j > 0$ among the coefficients f_i of the secret key $f(x)$. Suppose that the two consecutive coefficients f_i, f_{i-1} are both non-zero. We know their absolute values f'_i, f'_{i-1}. Choose and submit to the decryption oracle the polynomial $c(x) = \alpha|f_{i-1}| + \alpha|f_i|x$, with $\alpha \in (-q/2, q/2]$ such that $[2\alpha|f_{i-1} \cdot f_i|]_q \in \left[\frac{q}{4}, \frac{q}{2}\right]$ (it is always possible to find such an α). Now, the decryption oracle will compute and return the polynomial

$$D(c(x)) = \left[\left\lfloor \frac{2}{q}[\alpha|f_{i-1}|f_0 - \alpha|f_i|f_{n-1}]_q \right\rceil\right]_2 + \sum_{j=1}^{n-1} \left[\left\lfloor \frac{2}{q}[\alpha|f_{i-1}|f_j + \alpha|f_i|f_{j-1}]_q \right\rceil\right]_2 x^j$$

Let's focus on the i-th coefficient $\left[\left\lfloor \frac{2}{q}[\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1}]_q \right\rceil\right]_2$. We have two cases:

1) If f_i, f_{i-1} have different signs, then $\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1} = 0$, and therefore the i-th coefficient is zero $\left[\left\lfloor \frac{2}{q}[\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1}]_q \right\rceil\right]_2 = 0$

2) If f_i, f_{i-1} have the same positive sign, then $[\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1}]_q = [2\alpha|f_i f_{i-1}|]_q \in \left[\frac{q}{4}, \frac{q}{2}\right]$. In case f_i, f_{i-1} are both negative, we have that $[\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1}]_q = [-2\alpha|f_i f_{i-1}|]_q \in \left[-\frac{q}{2}, -\frac{q}{4}\right])$. In both cases, it easy to see that $\left[\left\lfloor \frac{2}{q}[\alpha|f_{i-1}|f_i + \alpha|f_i|f_{i-1}]_q \right\rceil\right]_2 = 1$

So we can distinguish whether two consecutive non-zero coefficients f_i, f_{i-1} have the same sign or not. Exactly as we saw for the attack on the [LATV12] scheme, this leads us to two possible candidates for the secret key; to determine which one is the correct one, it is enough to submit an extra appropriate query to the decryption oracle.

Remark 3. As we saw for the attack on the [LATV12] scheme, we have to be careful in case one of the coefficient f_i is zero. In this case in fact, no information can be given about the sign of f_{i-1} if we compare it to f_i. To solve this problem, we have to choose and submit to the decryption oracle a polynomial in the form $c(x) = a + bx^j$, for appropriates a, b, j. We omit the details, which are straightforward from what we have just discussed and from the attack on the [LATV12] scheme.

5 Conclusion

In this paper, we have described efficient key recovery attacks against the SHE schemes from [LATV12,BLLN13]. At this moment, it is still not clear whether we can adapt our attack to the scenario (3) of the LTV12 scheme, as noted in Remark 1 in the beginning of Section 3. This is an interesting future work. Up to today, the only known IND-CCA1 SHE scheme is that of Loftus et al. [LMSV12]. It is a wide open problem to design more efficient IND-CCA1 secure SHE schemes, possibly based on standard assumptions such as LWE.

Acknowledgements

Massimo Chenal is supported by an AFR PhD grant from the National Research Fund, Luxembourg. Qiang Tang is partially supported by a CORE (junior track) grant from the National Research Fund, Luxembourg. We thank the ePrint editors for pointing out references for three papers on key recovery attack against NTRUEncrypt.

References

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 309–325. ACM, 2012.

[BLLN13] Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *Cryptography and Coding*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012*, volume 7417 of *LNCS*, pages 868–886. 2012.

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Advances in Cryptology - CRYPTO 2011*, pages 505–524, 2011.

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. efficient fully homomorphic encryption from (standard) lwe. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 97–106, 2011.

[CT14] Massimo Chenal and Qiang Tang. On key recovery attacks against existing somewhat homomorphic encryption schemes. In *Progress in Cryptology - LATINCRYPT 2014*, volume 8895 of *Lecture Notes in Computer Science*, pages 239–258. Springer International Publishing, 2014.

[DGM15] Ricardo Dahab, Steven Galbraith, and Eduardo Morais. Adaptive key recovery attacks on ntru-based somewhat homomorphic encryption schemes. Cryptology ePrint Archive, Report 2015/127, 2015. http://eprint.iacr.org/.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178. ACM, 2009.

[GH11] Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. In *Advances in Cryptology - EUROCRYPT 2011*, pages 129–148, 2011.

[GN07] Nicolas Gama and PhongQ. Nguyen. New chosen-ciphertext attacks on ntru. In *Public Key Cryptography PKC 2007*. 2007.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and JuanA. Garay, editors, *Advances in Cryptology - CRYPTO 2013*, volume 8042 of *LNCS*, pages 75–92. 2013.

[HGNP$^+$03] Nick Howgrave-Graham, PhongQ. Nguyen, David Pointcheval, John Proos, JosephH. Silverman, Ari Singer, and William Whyte. The impact of decryption failures on the security of ntru encryption. In *Advances in Cryptology - CRYPTO 2003*, pages 226–246. 2003.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In JoeP. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer Berlin Heidelberg, 1998.

[JJ00] liane Jaulmes and Antoine Joux. A chosen-ciphertext attack against ntru. In *Advances in Cryptology CRYPTO 2000*, pages 20–35. Springer Berlin Heidelberg, 2000.

[LATV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 1219–1234, New York, NY, USA, 2012. ACM.

[LMSV12] Jake Loftus, Alexander May, Nigel P. Smart, and Frederik Vercauteren. On cca-secure somewhat homomorphic encryption. In *Proceedings of the 18th International Conference on Selected Areas in Cryptography*, SAC'11, pages 55–72, 2012.

[SS10] Damien Stehle and Ron Steinfeld. Faster fully homomorphic encryption. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 377–394. 2010.

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in Cryptology - EUROCRYPT 2010*, pages 24–43, 2010.

[ZPS12] Zhenfei Zhang, Thomas Plantard, and Willy Susilo. On the cca-1 security of somewhat homomorphic encryption over the integers. In *Proceedings of the 8th International Conference on Information Security Practice and Experience*, ISPEC'12, pages 353–368, 2012.