On CCA-Secure Fully Homomorphic Encryption

J. Loftus[1], A. May[2], N.P. Smart[1], and F. Vercauteren[3]

[1] Dept. Computer Science,
University of Bristol,
Merchant Venturers Building,
Woodland Road,
Bristol, BS8 1UB,
United Kingdom.
{loftus,nigel}@cs.bris.ac.uk

[2] Horst Görtz Institut für IT-Sicheiheit,
Ruhr-Universität Bochum.
Universtitätsstraße 150,
D-44780 Bochum,
Germany.
alex.may@rub.be

[3] COSIC - Electrical Engineering,
Katholieke Universiteit Leuven,
Kasteelpark Arenberg 10,
B-3001 Heverlee,
Belgium.
fvercaut@esat.kuleuven.ac.be

Abstract. It is well known that any encryption scheme which supports any form of homomorphic operation cannot be secure against adaptive chosen ciphertext attack. The question then arises as to what is the most stringent security definition which is achievable by homomorphic encryption schemes. Prior work has shown that various schemes which support a single homomorphic encryption scheme can be shown to be IND-CCA1, i.e. secure against lunchtime attacks. In this paper we extend this analysis to the recent fully homomorphic encryption scheme proposed by Gentry, as refined by Gentry, Halevi, Smart and Vercauteren. We show that the basic Gentry scheme is not IND-CCA1; indeed a trivial lunchtime attack allows one to recover the secret key. We then show that a minor modification to the variant of Smart and Vercauteren will allow one to achieve IND-CCA1, indeed PA-1, in the standard model assuming a lattice based knowledge assumption. We end by examining the security of the scheme against another security notion, namely security in the presence of ciphertext validity checking oracles.

1 Introduction

That some encryption schemes allow homomorphic operations, or exhibit so called *privacy homomorphisms* in the language of Rivest et. al [20], has often been considered a weakness. This is because any scheme which supports homomorphic operations is malleable, and hence is unable to achieve the de-facto security definition for encryption namely IND-CCA2. However, homomorphic encryption schemes do present a number of functional benefits. For example schemes which support a single additive homomorphic operation have been used to construct secure electronic voting schemes, e.g. [7, 10].

The usefulness of schemes supporting a single homomorphic operation has led some authors to consider what security definition existing homomorphic encryption schemes meet. A natural notion to try to achieve is that of IND-CCA1, i.e. security in the presence of a lunch-time attack. Lipmaa [18] shows that the ElGamal encryption scheme is IND-CCA1 secure with respect to a hard problem which is essentially the same as the IND-CCA1 security of the ElGamal scheme; a path of work recently extended in [2] to other schemes.

A different line of work has been to examine security in the context of Plaintext Awareness, introduced by Bellare and Rogaway [5] in the random oracle model and later refined into a hierarchy of security notions (PA-0, -1 and -2) by Bellare and Palacio [4]. Intuitively a scheme is said to be PA if the only way an adversary can create a valid ciphertext is by applying encryption to a public key and a valid message. Bellare and Palacio prove that a scheme which possesses both PA-1 (resp. PA-2) and is IND-CPA, is in fact secure against CCA1 (resp. CCA2) attacks.

The advantage of Bellare and Palacio's work is that one works in the standard model to prove security of a scheme; the disadvantage appears to be that one needs to make a strong assumption to prove a scheme is PA-1 or PA-2. The assumption required is a so-called *knowledge assumption*. That such a strong assumption is needed should not be surprising as the PA security notions are themselves very strong. In the context of encryption schemes supporting a single homomorphic operation Bellare and Pallacio show that the Cramer-Shoup Lite scheme [8] and an ElGamal variant introduced by Damgård [9] are both PA-1, and hence IND-CCA1, assuming the standard DDH (to obtain IND-CPA security) and a Diffie–Hellman knowledge assumption (to obtain PA-1 security). Informally, the Diffie–Hellman knowledge assumption is the assumption that an algorithm can only output a Diffie–Hellman tuple if the algorithm "knows" the discrete logarithm of one-tuple member with respect to another.

Rivest et. al originally proposed homomorphic encryption schemes so as to enable arbitrary computation on encrypted data. To perform such operations one would require an encryption scheme which supports two homomorphic operations, which are "complete" in the sense of allowing arbitrary computations. Such schemes are called fully homomorphic encryption (FHE) schemes, and it was not until Gentry's breakthrough construction in 2009 [13, 14] that such schemes could be constructed. Since Gentry's construction appeared a number of variants have been proposed, such as [12], as well as various simplifications [23] and improvements thereof [15]. All such schemes have been proved to be IND-CPA, i.e. secure under chosen plaintext attack.

At a high level all these constructions work in three stages: an initial *somewhat* homomorphic scheme which supports homomorphic evaluation of low degree polynomials, a process of squashing the decryption circuit and finally a bootstrapping procedure which will give fully homomorphic encryption and the evaluation of arbitrary functions on ciphertexts. In this paper we focus solely on the basic somewhat homomorphic scheme, but our attacks and analysis apply also to the extension using the bootstrapping process.

In this paper we consider the Smart–Vercauteren variant [23] of Gentry's scheme. In this variant there are two possible message spaces; one can either use the scheme to encrypt bits, and hence perform homomorphic operations in $\mathbb{F}_2$; or one can encrypt polynomials of degree N over $\mathbb{F}_2$. When one encrypts bits one achieves a scheme that is a specialisation of the original Gentry scheme, and it is this variant that has recently been realised by Gentry and Halevi [15]. We call this the Gentry–Halevi variant, to avoid confusion with other variants of Gentry's scheme, and we show that this scheme is not IND-CCA1 secure.

In particular we present a trivial complete break of the Gentry–Halevi variant scheme, in which the secret key can be recovered via a polynomial number of queries to a decryption oracle. The attack we propose works in a similar fashion to the attack of Bleichenbacher on RSA [6], in that on each successive oracle call we reduce the possible interval containing the secret key, based on the output of the oracle. Eventually the interval contains a single element, namely the secret key. Interesting all the Bleichenbacher style attacks on RSA, [6, 19, 22], recover a target message, and are hence strictly CCA2 attacks, whereas our attack takes no target ciphertext and recovers the key itself.

We then go on to show that a modification of the Smart–Vercauteren variant which encrypts polynomials can be shown to be PA-1, and hence is IND-CCA1. Informally we use the full Smart–Vercauteren variant to recover the random polynomial used to encrypt the plaintext polynomial in the decryption phase, and then we re-encrypt the result to check against the ciphertext. This forms a ciphertext validity check which then allows us to show PA-1 security based on a new lattice knowledge assumption. Our lattice knowledge assumption is a natural lattice based variant of the Diffie–Hellman knowledge assumption mentioned previously. In particular we assume that if an algorithm is able to output a non-lattice vector which is sufficiently close to a lattice vector then it must "know" the corresponding close lattice vector. We hope that this problem may be of independent interest in analysing other lattice based cryptographic schemes.

Finally, we end by examining possible extensions of the security notion for homomorphic encryption. We have remarked that a homomorphic encryption scheme (either one which supports single homomorphic operations or a FHE scheme) cannot be IND-CCA2, but we have examples of IND-CCA1 schemes. The question then arises as to whether IND-CCA1 is the "correct" security definition, i.e. whether this is the strongest definition one can obtain. In other contexts authors have considered attacks involving partial information oracles. In [11] Dent introduces the notion of a CPA+ attack, where the adversary is given access to an oracle which on input of a ciphertext outputs a single bit indicating whether the ciphertext is valid or not. Such a notion was originally introduced by Joye, Quisquater and Yung [17] in the context of attacking a variant of the EPOC-2 cipher which had been “proved” IND-CCA2. And was recently re-introduced under the name of a CVA (ciphertext verification) attack by Hu et al [16], in the context of symmetric encryption schemes. We use the term CVA rather than CPA+ as it conveys more easily the meaning of the security notion.

Such ciphertext validity oracles are actually the key component behind the traditional application of Bleichenbacher style attacks against RSA, in that one uses the oracle to recover information about the target plaintext. We show that our FHE scheme which is IND-CCA1 is not IND-CVA, by presenting an IND-CVA attack. The attack is not of the Bleichenbacher type, but is now more akin to the security reduction between search and decision LWE [21]. This attack opens up the possibility of a new FHE scheme which is also IND-CVA, a topic which we leave as an open problem.

Paper Summary: In Section 2 we recall standard definitions and notation. In Section 3 we discuss three variants of Gentry's scheme, all based on the simplification of Smart and Vercauteren. In Section 4 we describe our attack on the Gentry–Halevi variant. Then in Section 5 we introduce our lattice knowledge assumption and prove our third variant to be plaintext aware. Then in Section 6 we turn to discussing the security in the presence of ciphertext validity oracles.

2 Notation and Standard Definitions

For integers z, d reduction of z modulo d in the interval $[-d/2, d/2)$ will be denoted by $[z]_d$. For a rational number q, $\lfloor q \rceil$ will denote the rounding of q to the nearest integer, and $[q]$ denotes the (signed) distance between q and the nearest integer, i.e. $\lfloor q \rceil = q - [q]$. The notation $a \leftarrow b$ means assign the object b to a, whereas $a \leftarrow B$ for a set B means assign a uniformly at random from the set B. If B is an algorithm this means assign a with the output of B where the probability distribution is over the random coins of B.

For a polynomial $F(X) \in \mathbb{Z}[X]$ we let $\|F(X)\|_\infty$ denote the ∞-norm of the coefficient vector, i.e. the maximum coefficient in absolute value. If $F(X) \in \mathbb{Q}[X]$ then we let $\lfloor F(X) \rceil$ denote the polynomial in $\mathbb{Z}[X]$ obtained by rounding the coefficients of $F(X)$ to the nearest integer.

Fully Homomorphic Encryption: A fully homomorphic encryption scheme is a tuple of five algorithms $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add}, \mathsf{Mult})$ in which the message space is a ring $(R, +, \cdot)$ and the ciphertext space is also a ring $(\mathcal{R}, \oplus, \otimes)$ such that for all messages $m_1, m_2 \in R$, and all outputs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, we have

$$m_1 + m_2 = \mathsf{Decrypt}(\mathsf{Encrypt}(m_1, \mathsf{pk}) \oplus \mathsf{Encrypt}(m_2, \mathsf{pk}), \mathsf{sk})$$
$$m_1 \cdot m_2 = \mathsf{Decrypt}(\mathsf{Encrypt}(m_1, \mathsf{pk}) \otimes \mathsf{Encrypt}(m_2, \mathsf{pk}), \mathsf{sk}).$$

Security Notions for Public Key Encryption: For a public key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ (whether standard, homomorphic, or fully homomorphic) semantic security is captured by the following game between a challenger and an adversary $\mathcal{A}$;

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

The adversary is said to win the game if $b = b'$, with the advantage of the adversary winning the game being defined by

$$\mathrm{Adv}_{A,\mathcal{E},\lambda}^{IND-xxx} = |\Pr(b = b') - 1/2|\,.$$

A scheme is said to be IND-xxx secure if no polynomial time adversary $\mathcal{A}$ can win the above game with non-negligible advantage in the security parameter λ. The precise security notion one obtains depends on the oracle access one gives the adversary in its different stages.

- If $\mathcal{A}$ has access to no oracles in either stage then xxx=CPA.
- If $\mathcal{A}$ has access to a decryption oracle in stage one then xxx=CCA1.
- If $\mathcal{A}$ has access to a decryption oracle in both stages then xxx=CCA2, often now denoted simply CCA.
- If $\mathcal{A}$ has access to a ciphertext validity oracle in both stages, which on input of a ciphertext determines whether it would output $\perp$ or not on decryption, then xxx=CVA.

Looking ahead to the end of the paper, one open problem is whether there exists an FHE scheme that is both IND-CCA1 and IND-CVA.

LATTICES: A (full-rank) lattice is simply a discrete subgroup of $\mathbb{R}^n$ generated by n linear independent vectors, $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, called a basis. Every lattice has an infinite number of bases, with each set of basis vectors being related by a unimodular transformation matrix. If B is such a set of vectors, we write

$$L = \mathcal{L}(B) = \{\mathbf{v} \cdot B | \mathbf{v} \in \mathbb{Z}^n\}$$

to be the resulting lattice. An integer lattice is a lattice in which all the bases vectors have integer coordinates.

For any basis there is an associated fundamental parallelepiped which can be taken as $\mathcal{P}(B) = \{\sum_{i=1}^n x_i \cdot \mathbf{b}_i | x_i \in [-1/2, 1/2)\}$. The volume of this fundamental parallelepiped is given by the absolute value of the determinant of the basis matrix $\Delta = |\det(B)|$. We denote by $\lambda_\infty(L)$ the ∞-norm of a shortest vector (for the ∞-norm) in L.

3 The Smart-Vercauteren Variant of Gentry's Scheme

We will be examining variants of Gentry's FHE scheme [13], in particular three variants based on the simplification of Smart and Vercauteren [23], as optimized by Gentry and Halevi [15]. All variants make use of the same key generation procedure, parametrized by a tuple of integers (N, t, μ); we assume there is a function mapping security parameters λ into tuples (N, t, μ). In practice N will be a power of two, t will be greater than $2^{\sqrt{N}}$ and μ will be a small integer, perhaps one.

$\mathsf{KeyGen}(1^\lambda)$
- Pick an irreducible polynomial $F \in \mathbb{Z}[X]$ of degree N.
- Pick a polynomial $G(X) \in \mathbb{Z}[X]$ of degree at most $N-1$, with coefficients bounded by t.
- $d \leftarrow \mathsf{resultant}(F, G)$.
- G is chosen such that $G(X)$ has a single unique root in common with $F(X)$ modulo d. Let α denote this root.
- $Z(X) \leftarrow d/G(X) \pmod{F(X)}$.
- $\mathsf{pk} \leftarrow (\alpha, d, \mu, F(X))$, $\mathsf{sk} \leftarrow (Z(X), G(X), d, F(X))$.

In [15] Gentry and Halevi show how to compute, for the polynomial $F(X) = X^{2^n} + 1$, the root α and the polynomial $Z(X)$ using a method based on the Fast Fourier Transform. In particular they show how this can be done for non-prime values of d (removing one of the main restrictions in the key generation method proposed in [23]).

By construction, the principal ideal $\mathfrak{g}$ generated by $G(X)$ in the number field $K = \mathbb{Z}[X]/(F(X))$ is equal to the ideal with $\mathcal{O}_K$ basis $(d, X - \alpha)$. In particular, the ideal $\mathfrak{g}$ precisely consists of all elements in $\mathbb{Z}[X]/(F(X))$ that are zero when evaluated at α modulo d. The Hermite-Normal-Form of a basis matrix of the lattice defined by the coefficient vectors of $\mathfrak{g}$ is simply given by

$$B = \begin{pmatrix} d & & & 0 \\ -\alpha & 1 & & \\ -\alpha^2 & & 1 & \\ \vdots & & & \ddots \\ -\alpha^{N-1} & 0 & & 1 \end{pmatrix}, \tag{1}$$

where the elements in the first column are reduced modulo d.

To aid what follows we write $Z(X) = z_0 + z_1 \cdot X + \ldots + z_{N-1} \cdot X^{N-1}$ and define

$$\delta_\infty = \sup \left\{ \frac{\|g(X) \cdot h(X) \pmod{F(X)}\|_\infty}{\|g(X)\|_\infty \cdot \|h(X)\|_\infty} : g, h \in \mathbb{Z}[X], \deg(g), \deg(h) < N \right\}.$$

For the choice $f = x^N + 1$, we have $\delta_\infty = N$. The key result to understand how the simplification of Smart and Vercauteren to Gentry's scheme works is the following lemma adapted from [23].

Lemma 1. *Let $Z(X), G(X), \alpha$ and d be as defined in the above key generation procedure. If $C(X) \in \mathbb{Z}[X]/(F(X))$ is a polynomial with $\|C(X)\|_\infty < U$ and set $c = C(\alpha) \pmod d$, then*

$$C(X) = c - \left\lfloor \frac{c \cdot Z(X)}{d} \right\rceil \cdot G(X) \pmod{F(X)}$$

for

$$U = \frac{d}{2 \cdot \delta_\infty \cdot \|Z(X)\|_\infty} .$$

Proof. By definition of c, we have that $c - C(X)$ is contained in the principal ideal generated by $G(X)$ and thus there exists a $q(X) \in \mathbb{Z}[X]/(F(X))$ such that $c - C(X) = q(X)G(X)$. Using $Z(X) = d/G(X) \pmod{F(X)}$, we can write

$$q(X) = \frac{cZ(X)}{d} - \frac{C(X)Z(X)}{d} .$$

Since $q(X)$ has integer coefficients, we can recover it by rounding the coefficients of the first term if the coefficients of the second term are strictly bounded by $1/2$. This shows that $C(X)$ can be recovered from c for $\|C(X)\|_\infty < d/(2\delta_\infty \|Z(X)\|_\infty)$.

Note that the above lemma essentially states that if $\|C(X)\|_\infty < U$, then $C(X)$ is determined uniquely by its evaluation in α modulo d. Recall that any polynomial $H(X)$ of degree less than $N - 1$, whose coefficient vector is in the lattice defined in equation (1), satisfies $H(\alpha) = 0 \pmod d$. Therefore, if $H(X) \neq 0$, the lemma implies, for such an H, that $\|H(X)\|_\infty \geq T$, and thus we conclude that $U \leq \lambda_\infty(L)$. Since $G(X)$ clearly has a coefficient vector in the lattice L, we conclude that

$$U \leq \lambda_\infty(L) \leq \|G(X)\|_\infty .$$

Although Lemma 1 provides the maximum value of U for which ciphertexts are decryptable, we will only allow half of this maximum value, i.e. $T = U/4$. As such we are guaranteed that $T \leq \lambda_\infty(L)/4$. We note that T defines the size of the circuit that the somewhat homomorphic encryption scheme can deal with.

Using the above key generation method we can define three variants of the Smart–Vercauteren variant of Gentry's scheme. The first variant is the one used in the Gentry/Halevi implementation of [15], the second is the general variant proposed by Smart and Vercauteren, whereas the third divides the decryption procedure into two steps and provides a ciphertext validity check. In later sections we shall show that the first variant is not IND-CCA1 secure, and by extension neither is the second variant. However, we will show that the third variant is indeed IND-CCA1. We will then show that the third variant is not IND-CVA secure.

Each of the following variants is only a somewhat homomorphic scheme, extending it to a fully homomorphic scheme can be performed using methods of [13–15] without affecting any of the results in this paper.

Gentry–Halevi Variant: The plaintext space is the field $\mathbb{F}_2$. The above KeyGen algorithm is modified to only output keys for which $d \equiv 1 \pmod 2$. This implies that at least one coefficient of $Z(X)$, say z_{i_0} will be odd. We replace $Z(X)$ in the private key with z_{i_0}, and can drop the values $G(X)$ and $F(X)$ entirely from the private key. Encryption and decryption can now be defined via the functions:

Encrypt$(m, \mathsf{pk}; r)$	Decrypt(c, sk)
– $R(X) \leftarrow \mathbb{Z}[X]$ s.t. $\|R(X)\|_\infty \leq \mu$.	– $m \leftarrow [c \cdot z_{i_0}]_d \pmod 2$
– $C(X) \leftarrow m + 2 \cdot R(X)$.	– Return m.
– $c \leftarrow [C(\alpha)]_d$.	
– Return c.	

Full-Space Smart–Vercauteren: In this variant the plaintext space is the algebra $\mathbb{F}_2[X]/(F(X))$, where messages are given by binary polynomials of degree less than N. As such we call this the Full-Space Smart–Vercauteren system as the plaintext space is the full set of binary polynomials. We modify the above key generation algorithm so that it only outputs keys for which the polynomial $G(X)$ satisifies $G(X) \equiv 1 \pmod 2$. This results in algorithms defined by:

Encrypt$(M(X), \mathsf{pk}; r)$	Decrypt(c, sk)
– $R(X) \leftarrow \mathbb{Z}[X]$ s.t. $\|R(X)\|_\infty \leq \mu$.	– $C(X) \leftarrow c - \lfloor c \cdot Z(X)/d \rceil$.
– $C(X) \leftarrow M(X) + 2 \cdot R(X)$.	– $M(X) \leftarrow C(X) \pmod 2$.
– $c \leftarrow [C(\alpha)]_d$.	– Return $M(X)$.
– Return c.	

That decryption works, assuming the input ciphertext corresponds to the evaluation of a polynomial with coefficients bounded by T, follows from Lemma 1 and the fact that $G(X) \equiv 1 \pmod 2$.

ccFHE: This is our ciphertext-checking FHE scheme (or ccFHE scheme for short). This is exactly like the above Full-Space Smart–Vercauteren variant in terms of key generation, but we now check the ciphertext before we output the message. Thus encryption/decryption become;

Encrypt$(M(X), \mathsf{pk}; r)$	Decrypt(c, sk)
– $R(X) \leftarrow \mathbb{Z}[X]$ s.t. $\|R(X)\|_\infty \leq \mu$.	– $C(X) \leftarrow c - \lfloor c \cdot Z(X)/d \rceil \cdot G(X)$.
– $C(X) \leftarrow M(X) + 2 \cdot R(X)$.	– $C(X) \leftarrow C(X) \pmod{F(X)}$
– $c \leftarrow [C(\alpha)]_d$.	– $c' \leftarrow [C(\alpha)]_d$.
– Return c.	– If $c' \neq c$ or $\|C(X)\|_\infty \geq T$ return $\perp$.
	– $M(X) \leftarrow C(X) \pmod 2$.
	– Return $M(X)$.

4 CCA1 attack on the Gentry–Halevi Variant

We construct a CCA1 attacker against the above Gentry–Halevi variant. Let z be the secret key, i.e. the specific odd coefficient of $Z(X)$ chosen by the decryptor. Note that we can assume $z \in [0, d)$, since decryption in the Gentry–Halevi variant works for any secret key $z + k \cdot d$ with $k \in \mathbb{Z}$. We assume the attacker has access to a decryption oracle to which it can make polynomially many queries, $\mathcal{O}_D(c)$. On each query the oracle returns the value of $[c \cdot z]_d \pmod 2$.

In Algorithm 1 we present pseudo-code to describe how the attack proceeds. We start with an interval $[L, \ldots, U]$ which is known to contain the secret key z and in each iteration we split the interval into two halves determined by a specific ciphertext c. The choice of which sub-interval to take next depends on whether k multiples of d are sufficient to reduce $c \cdot z$ into the range $[-d/2, \dots, d/2)$ or whether $k+1$ multiples are required.

Algorithm 1: CCA1 attack on the Gentry–Halevi Variant

$L \leftarrow 0, U \leftarrow d-1$
while $U - L > 1$ **do**
  $c \leftarrow \lfloor d/(U-L) \rfloor$
  $b \leftarrow \mathcal{O}_D(c)$
  $q \leftarrow (c+b) \mod 2$
  $k \leftarrow \lfloor Lc/d + 1/2 \rfloor$
  $B \leftarrow (k+1/2)d/c$
  if $(k \bmod 2 = q)$ **then**
    $U \leftarrow \lfloor B \rfloor$
  else
    $L \leftarrow \lceil B \rceil$
return L

<u>Analysis</u>: The core idea of the algorithm is simple: in each step we choose a "ciphertext" c such that the length of the interval for the quantity $c \cdot z$ is bounded by d. Since in each step, $z \in [L, U]$, we need to take $c = \lfloor d/(U-L) \rfloor$. As such it is easy to see that $c(U-L) \leq d$.

To reduce cL, we need to subtract kd such that $-d/2 \leq cL - kd < d/2$, which shows that $k = \lfloor Lc/d + 1/2 \rfloor$. Furthermore, since the length of the interval for $c \cdot z$ is bounded by d, there will be exactly one boundary of the form $d/2 + id$ in $[cL, cU]$, namely $d/2 + kd$. This means that there is exactly one boundary $B = (k+1/2)d/c$ in the interval for z.

Define q as the unique integer such that $-d/2 \leq cz - qd < d/2$, then since the length of the interval for $c \cdot z$ is bounded by d, we either have $q = k$ or $q = k+1$. To distinguish between the two cases, we simply look at the output of the decryption oracle: recall that the oracle outputs $[c \cdot z]_d \pmod 2$, i.e. the bit output by the oracle is

$$b = c \cdot z - q \cdot d \pmod 2 = (c + q) \pmod 2 .$$

Therefore, $q = (b + c) \pmod 2$ which allows us to choose between the cases k and $k+1$. If $q = k \pmod 2$, then z lies in the first part $[L, \lfloor B \rfloor]$, whereas in the other case, z lies in the second part $[\lceil B \rceil, U]$.

Having proved correctness we now estimate the running time. The behaviour of the algorithm is easily seen to be as follows: in each step, we obtain a boundary B in the interval $[L, U]$ and the next interval becomes either $[L, \lfloor B \rfloor]$ or $[\lceil B \rceil, U]$. Since B can be considered random in $[L, U]$ as well as the choice of the interval, this shows that in each step, the size of the interval decreases by a factor 2 on average. In conclusion we deduce that recovering the secret key will require $O(\log d)$ calls to the oracle.

5 ccFHE is PA-1

In this section we prove that the ccFHE encryption scheme given earlier is PA-1, assuming a lattice knowledge assumption holds. We first recap on the definition of PA-1 in the standard model, and then we introduce our lattice knowledge assumption. Once this is done we present the proof.

PLAINTEXT AWARENESS – PA-1: The original intuition for introducing plaintext awareness was as follows - if an adversary knowns the plaintext corresponding to every ciphertext it produces, then the adversary has no need for a decryption oracle and hence, PA+IND-CPA must imply IND-CCA. However, there are subtleties in the definition for plaintext awareness, leading to three definitions, PA-0, PA-1 and PA-2. However, after suitably formalizing the definitions, PA-x plus IND-CPA implies IND-CCAx, for x = 1 and 2. In our context we are only interested in CCA1 security, so we will only discuss the notion of PA-1 in this paper.

Before formalising this it is worth outlining some of the terminology. We have a polynomial time adversary $\mathcal{A}$ called a *ciphertext creator*, that takes as input a public key and can query ciphertexts to an oracle. An algorithm $\mathcal{A}^*$ is called a *successful extractor* for $\mathcal{A}$ if it can provide responses to $\mathcal{A}$ which are computationally indistinguishable from those provided by a decryption oracle. In particular a scheme is said to be PA-1 if there exists a successful extractor for any ciphertext creator that makes a polynomial number of queries. The extractor gets the same public key as $\mathcal{A}$ and also has access to the random coins used by algorithm $\mathcal{A}$. Following [4] we define PA-1 formally as follows:

Definition 1 (PA1). *Let $\mathcal{E}$ be a public key encryption scheme and $\mathcal{A}$ be an algorithm with access to an oracle $\mathcal{O}$ taking input* pk *and returning a string. Let $\mathcal{D}$ be an algorithm that takes as input a string and returns a single bit and let $\mathcal{A}^*$ be an algorithm which takes as input a string and some state information and returns either a string or the symbol $\perp$, plus a new state. We call $\mathcal{A}$ a* ciphertext creator*, $\mathcal{A}^*$ a* PA-1-extractor*, and $\mathcal{D}$ a* distinguisher*. For security parameter λ we define the experiments in Figure 1, defining the PA-1 advantage to be*

$$\mathrm{Adv}^{PA\text{-}1}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda) = \left|\Pr(\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda) = 1) - \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda) = 1)\right|.$$

We say $\mathcal{A}^*$ is a successful PA-1-extractor for $\mathcal{A}$, if for every polynomial time distinguisher the above advantage is negligible.

Note, in experiment $\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda)$ the algorithm $\mathcal{A}$'s oracle queries are responded to by the genuine decryption algorithm, whereas in $\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{A}^*}(\lambda)$ the queries are responded to by the PA-1-extractor. If $\mathcal{A}^*$ did not receive the coins $\mathsf{cc}[\mathcal{A}]$ from $\mathcal{A}$ then it would be functionally equivalent to the real decryption oracle, thus the fact that $\mathcal{A}^*$ gets access to the coins in the second experiment is crucial. Also note that the distinguisher acts independently of $\mathcal{A}^*$, and thus this is strictly stronger than having $\mathcal{A}$ decide as to whether it is interacting with an extractor or a real decryption oracle.

$\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda)$:

- $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
- $x \leftarrow \mathcal{A}^{\mathsf{Decrypt}(\cdot,\mathsf{sk})}(\mathsf{pk})$.
- $d \leftarrow \mathcal{D}(x)$.
- Return d.

$\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda)$:

- $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
- Choose coins $\mathsf{cc}[\mathcal{A}]$ (resp. $\mathsf{cc}[\mathcal{A}^*]$) for $\mathcal{A}$ (resp. $\mathcal{A}^*$).
- $\mathsf{St} \leftarrow (\mathsf{pk}, \mathsf{cc}[\mathcal{A}])$.
- $x \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk};\mathsf{cc}[\mathcal{A}])$, replying to the oracle queries $\mathcal{O}(c)$ as follows:
  - $(m, \mathsf{St}) \leftarrow \mathcal{A}^*(c, \mathsf{St}; \mathsf{cc}[\mathcal{A}^*])$.
  - Return m to $\mathcal{A}$
- $d \leftarrow \mathcal{D}(x)$.
- Return d.

Fig. 1. Experiments $\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}$ and $\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{A}^*}$

The intuition is that $\mathcal{A}^*$ acts as the unknowing subconscious of $\mathcal{A}$, and is able to extract knowledge about $\mathcal{A}$'s queries to its oracle. That $\mathcal{A}^*$ can obtain the underlying message captures the notion that $\mathcal{A}$ needs to know the message before it can output a valid ciphertext.

The following lemma is taken from [4] and will be used in the proof of the main theorem.

Lemma 2. *Let $\mathcal{E}$ be a public key encryption scheme. Let $\mathcal{A}$ be a polynomial-time ciphertext creator attacking $\mathcal{E}$, $\mathcal{D}$ a polynomial-time distinguisher, and $\mathcal{A}^*$ a polynomial-time PA-1-extractor. Let* DecOK *denote the event that all $\mathcal{A}^*$'s answers to $\mathcal{A}$'s queries are correct in experiment $\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda)$. Then,*

$$\Pr(\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda) = 1) \geq \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda) = 1) - \Pr(\overline{\mathsf{DecOK}})$$

<u>Lattice Knowledge Assumption</u>: Our knowledge assumption can be stated informally as follows: suppose there is a (probabilistic) algorithm $\mathcal{C}$ which takes as input a lattice basis of a lattice L and outputs a vector $\mathbf{c}$ suitably close to a lattice point $\mathbf{p}$, i.e. closer than $\epsilon \cdot \lambda_\infty(L)$ in the ∞-norm for a fixed $\epsilon \in (0, 1/2)$. Then there is an algorithm $\mathcal{C}^*$ which on input of $\mathbf{c}$ and the random coins of $\mathcal{C}$ outputs a close lattice vector $\mathbf{p}$, i.e. one for which $\|\mathbf{c} - \mathbf{p}\|_\infty < \epsilon \cdot \lambda_\infty(L)$. Note that the algorithm $\mathcal{C}^*$ can therefore act as a ϵ-CVP-solver for $\mathbf{c}$ in the ∞-norm, given the coins $\mathsf{cc}[\mathcal{C}]$. Again as in the PA-1 definition it is perhaps useful to think of $\mathcal{C}^*$ as the "subconscious" of $\mathcal{C}$, since $\mathcal{C}$ is capable of outputting a vector close to the lattice it must have known the close lattice vector in the first place. Formally we have:

Definition 2 (LK-ϵ). *Let ϵ be a fixed constant in the interval $(0, 1/2)$. Let $\mathcal{G}$ denote an algorithm which on input of a security parameter 1^λ outputs a lattice L given by a basis B of dimension $n = n(\lambda)$ and volume $\Delta = \Delta(\lambda)$. Let $\mathcal{C}$ be an algorithm that takes a lattice basis B as input, and has access to an oracle $\mathcal{O}$, and returns nothing. Let $\mathcal{C}^*$ denote an algorithm which takes as input a vector*

$\mathbf{c} \in \mathbb{R}^n$ and some state information, and returns another vector $\mathbf{p} \in \mathbb{R}^n$ plus a new state. Consider the experiment in Figure 2. The LK-ϵ advantage of $\mathcal{C}$ relative to $\mathcal{C}^*$ is defined by

$$\mathrm{Adv}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda) = \Pr[\mathrm{Exp}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda) = 1].$$

We say G satisfies the LK-ϵ assumption, for a fixed ϵ, if for every polynomial time $\mathcal{C}$ there exists a polynomial time $\mathcal{C}^*$ such that $\mathrm{Adv}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)$ is a negligible function of λ.

$\mathrm{Exp}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)$:

- $B \leftarrow \mathcal{G}(1^\lambda)$.
- Choose coins $\mathsf{cc}[\mathcal{C}]$ (resp. $\mathsf{cc}[\mathcal{C}^*]$) for $\mathcal{C}$ (resp. $\mathcal{C}^*$).
- $\mathsf{St} \leftarrow (B, \mathsf{cc}[\mathcal{C}])$.
- Run $\mathcal{C}^{\mathcal{O}}(B; \mathsf{cc}[\mathcal{C}])$ until it halts, replying to the oracle queries $\mathcal{O}(\mathbf{c})$ as follows:
  - $(\mathbf{p}, \mathsf{St}) \leftarrow \mathcal{C}^*(\mathbf{c}, \mathsf{St}; \mathsf{cc}[\mathcal{C}^*])$.
  - If $\mathbf{c}$ is not within distance $\epsilon \cdot \lambda_\infty(L)$ (w.r.t. the ∞-norm) of the lattice, then return $\mathbf{p}$ to $\mathcal{C}$.
  - If $\mathbf{p} \notin \mathcal{L}(B)$, return 1.
  - If $\|\mathbf{p} - \mathbf{c}\|_\infty > \epsilon \cdot \lambda_\infty(L)$, return 1.
  - Return $\mathbf{p}$ to $\mathcal{C}$.
- Return 0.

Fig. 2. Experiment $\mathrm{Exp}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)$

The algorithm $\mathcal{C}$ is called an LK-ϵ adversary and $\mathcal{C}^*$ a LK-ϵ extractor. We now discuss this assumption in more detail. Notice, that for all lattices that if $\epsilon < 1/4$ then the probability of a random vector being with $\epsilon \cdot \lambda_\infty(L)$ of the lattice is bounded from above by $1 - 1/2^n$, and for lattices which are not highly orthogonal this is likely to hold for all ϵ up to 1/2. Our choice of T in the ccFHE scheme as $U/4$ is to guarantee that our lattice knowledge assumption is applied with $\epsilon = 1/4$, and hence is more likely to hold.

If the query $\mathbf{c}$ which $\mathcal{C}$ asks of its oracle is within $\epsilon \cdot \lambda_\infty(L)$ of a lattice point then we require that $\mathcal{C}^*$ finds such a close lattice point. If it does not then the experiment will output 1; and the assumption is that this happens with negligible probability.

Notice that if $\mathcal{C}$ asks its oracle a query of a vector which is not within $\epsilon \cdot \lambda_\infty(L)$ of a lattice point then the algorithm $\mathcal{C}^*$ may do whatever it wants. However, to determine this condition within the experiment we require that the environment running the experiment is all powerful, in particular, that it can compute $\lambda_\infty(L)$ and decide whether a vector is close enough to the lattice. Thus our experiment, but not algorithms $\mathcal{C}$ and $\mathcal{C}^*$, is assumed to be information theoretic. This might seem strange at first sight but is akin to a similarly powerful game experiment in the strong security model for certificateless encryption [1], or the definition of insider unforgeable signcryption in [3].

For certain input bases, e.g. reduced ones or ones of small dimension, an algorithm $\mathcal{C}^*$ can be constructed by standard algorithms to solve the CVP problem. This does not contradict our assumption, since $\mathcal{C}$ would also be able to apply such an algorithm and hence "know" the close lattice point. Our assumption is that when this is not true, the only way $\mathcal{C}$ could generate a close lattice point (for small enough values of ϵ) is by computing $\mathbf{x} \in \mathbb{Z}^n$ and perturbing the vector $\mathbf{x} \cdot B$.

MAIN THEOREM:

Theorem 1. *Let $\mathcal{G}$ denote the lattice basis generator induced from the* KeyGen *algorithm of the ccFHE scheme, i.e. for a given security parameter 1^λ, run* $\mathsf{KeyGen}(1^\lambda)$ *to obtain* $\mathsf{pk} = (\alpha, d, \mu, F(X))$ *and* $\mathsf{sk} = (Z(X), G(X), d, F(X))$, *and generate the lattice basis B as in equation* (1). *Then, if $\mathcal{G}$ satisfies the LK-ϵ assumption for $\epsilon = 1/4$ then the ccFHE scheme is PA-1.*

Proof. Let $\mathcal{A}$ be a polynomial-time ciphertext creator attacking the ccFHE scheme, then we show how to construct a polynomial time PA1-extractor $\mathcal{A}^*$. The creator $\mathcal{A}$ takes as input the public key $\mathsf{pk} = (\alpha, d, \mu, F(X))$ and random coins $\mathsf{cc}[\mathcal{A}]$ and returns an integer as the candidate ciphertext. To define $\mathcal{A}^*$, we will exploit $\mathcal{A}$ to build a polynomial-time LK-ϵ adversary $\mathcal{C}$ attacking the generator $\mathcal{G}$. By the LK-ϵ assumption there exists a polynomial-time LK-ϵ extractor $\mathcal{C}^*$, that will serve as the main building block for the PA1-extractor $\mathcal{A}^*$. The description of the LK-ϵ adversary $\mathcal{C}$ is given in Figure 3 and the description of the PA-1-extractor $\mathcal{A}^*$ is given in Figure 4.

LK-ϵ adversary $\mathcal{C}^{\mathcal{O}}(B; \mathsf{cc}[\mathcal{C}])$
- Let $d = B[0][0]$ and $\alpha = -B[1][0]$
- Parse $\mathsf{cc}[\mathcal{C}]$ as $\mu || F(X) || \mathsf{cc}[\mathcal{A}]$
- Run $\mathcal{A}$ on input $(\alpha, d, \mu, F(X))$ and coins $\mathsf{cc}[\mathcal{A}]$ until it halts, replying to its oracle queries as follows:
  - If $\mathcal{A}$ makes a query with input c, then
  - Submit $(c, 0, 0, \dots, 0)$ to $\mathcal{O}$ and let $\mathbf{p}$ denote the response
  - Let $\mathbf{c} = (c, 0, \dots, 0) - \mathbf{p}$, and $C(X) = \sum_{i=0}^{N-1} \mathbf{c}_i X^i$
  - Let $c' = [C(\alpha)]_d$
  - If $c' \neq c$ or $\|C(X)\|_\infty \geq T$, then $M(X) \leftarrow \perp$, else $M(X) \leftarrow C(X) \pmod 2$
  - Return $M(X)$ to $\mathcal{A}$ as the oracle response.
- Halt

Fig. 3. LK-ϵ adversary

We first show that $\mathcal{A}^*$ is a successful PA-1-extractor for $\mathcal{A}$. In particular, let DecOK denote the event that all $\mathcal{A}^*$'s answers to $\mathcal{A}$'s queries are correct in experiment $\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathrm{ccFHE},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda)$, then we have that $\Pr(\overline{\mathsf{DecOK}}) \leq \mathrm{Adv}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)$.

We first consider the case that c is a valid ciphertext, i.e. a ciphertext such that $\mathsf{Decrypt}(c, \mathsf{sk}) \neq \perp$, then by definition of Decrypt in the ccFHE scheme there

PA-1-extractor $\mathcal{A}^*(c, \mathsf{St}[\mathcal{A}^*]; \mathsf{cc}[\mathcal{A}^*])$

- If $\mathsf{St}[\mathcal{A}^*]$ is initial state then
  - parse $\mathsf{cc}[\mathcal{A}^*]$ as $(\alpha, d, \mu, F(X))||\mathsf{cc}[\mathcal{A}]$
  - $\mathsf{St}[C^*] \leftarrow (\alpha, d, \mu, F(X))||\mathsf{cc}[\mathcal{A}]$
  - else parse $\mathsf{cc}[\mathcal{A}^*]$ as $(\alpha, d, \mu, F(X))||\mathsf{St}[\mathcal{C}^*]$
- $(\mathbf{p}, \mathsf{St}[\mathcal{C}^*]) \leftarrow \mathcal{C}^*((c, 0, \ldots, 0), \mathsf{St}[\mathcal{C}^*]; \mathsf{cc}[\mathcal{A}^*])$
- Let $\mathbf{c} = (c, 0, \ldots, 0) - \mathbf{p}$, and $C(X) = \sum_{i=0}^{N-1} \mathbf{c}_i X^i$
- Let $c' = [C(\alpha)]_d$
- If $c' \neq c$ or $\|C(X)\|_\infty \geq T$, then $M(X) \leftarrow \perp$, else $M(X) \leftarrow C(X) \pmod 2$
- $\mathsf{St}[\mathcal{A}^*] \leftarrow (\alpha, d, \mu, F(X))||\mathsf{St}[C^*]$
- Return $(M(X), \mathsf{St}[\mathcal{A}^*])$.

Fig. 4. PA-1-extractor

exists a $C(x)$ such that $c = [C(\alpha)]_d$ and $\|C(X)\|_\infty \leq T$. Let $\mathbf{p}'$ be the coefficient vector of $c - C(X)$, then by definition of c, we have that $\mathbf{p}'$ is a lattice vector that is within distance T of the vector $(c, 0, \ldots, 0)$. Furthermore, since $T \leq \lambda_\infty(L)/4$, the vector $\mathbf{p}'$ is the *unique* vector with this property. Let $\mathbf{p}$ be the vector returned by $\mathcal{C}^*$ and assume that $\mathbf{p}$ passes the test $\|(c, 0, \ldots, 0) - \mathbf{p}\|_\infty \leq T$, then we conclude that $\mathbf{p} = \mathbf{p}'$. This shows that if c is a valid ciphertext, it will be decrypted correctly by $\mathcal{A}^*$.

When c is an invalid ciphertext then the real decryption oracle will always output $\perp$, and it can be easily seen that our PA-1 extractor $\mathcal{A}^*$ will also output $\perp$. Thus in the case of an invalid ciphertext the adversary $\mathcal{A}$ cannot tell the two oracles apart.

The theorem now follows from combining $\Pr(\overline{\mathsf{DecOK}}) \leq \mathrm{Adv}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)$ with Lemma 2 as follows:

$$\begin{aligned}\mathrm{Adv}^{PA\text{-}1}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda) &= \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda) = 1) - \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}x}_{\mathcal{E},\mathcal{A},\mathcal{D},\mathcal{A}^*}(\lambda) = 1) \\ &\leq \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda) = 1) - \Pr(\mathrm{Exp}^{PA\text{-}1\text{-}d}_{\mathcal{E},\mathcal{A},\mathcal{D}}(\lambda) = 1) + \Pr(\overline{\mathsf{DecOK}}) \\ &\leq \mathrm{Adv}^{LK\text{-}\epsilon}_{\mathcal{G},\mathcal{C},\mathcal{C}^*}(\lambda)\,.\end{aligned}$$

6 ccFHE is not secure in the presence of a CVA attack

We now show that our ccFHE scheme is not secure when the attacker, after being given the target ciphertext c^*, is given access to an oracle $\mathcal{O}_{CVA}(c)$ which returns 1 if c is a valid ciphertext (i.e. the decryption algorithm would output a message), and which returns 0 if it is invalid (i.e. the decryption algorithm would output $\perp$). Such an "oracle" can often be obtained in the real world by the attacker observing the behaviour of a party who is fed ciphertexts of the attackers choosing. Since a CVA attack is strictly weaker than a CCA2 attack and strictly stronger that a CCA1 attack it is an interesting open (and practical) question as to whether an FHE scheme can be CVA secure.

We now show that the ccFHE scheme is not CVA secure, by presenting a relatively trivial attack: Suppose the adversary is given a target ciphertext c^*

associated with a hidden message m^*. Using the method in Algorithm 2 it is easy to determine the message using access to $\mathcal{O}_{CVA}(c)$. Basically, we add on multiples of α^i to the ciphertext until it does not decrypt; this allows us to perform a binary search on the coefficient of $C(X)$ at i, since we know the bound T on the coefficients of $C(X)$.

Algorithm 2: CVA attack on ccFHE

```
C(X) ← 0
for i from 0 upto N − 1 do
    L ← −T + 1, U ← T − 1
    while U ≠ L do
        M ← ⌈(U + L)/2⌉.
        c ← [−c* + (M + T − 1) · α^i]_d.
        if O_CVA(c) = 1 then
            L ← M.
        else
            U ← M − 1.
    C(X) ← C(X) + U · X^i.
m* ← C(X) (mod 2)
return m*
```

If c_i is the ith coefficient of the actual $C(X)$ underlying the target ciphertext c^*, then the ith coefficient of the polynomial underlying ciphertext c being passed to the $\mathcal{O}_{CVA}$ oracle is given by $M + T - 1 + c_i$. When $M \leq c_i$ this coefficient is less than T and so the oracle will return 1, however when $M > c_i$ the coefficient is greater than or equal T and hence the oracle will return 0. Thus we can divide the interval for c_i in two depending on the outcome of the test.

It is obvious that the complexity of the attack is $O(N \cdot \log_2 T)$. Since, for the recommended parameters in the key generation method, N and $\log_2 T$ are a polynomial functions of the security parameter, we obtain a polynomial time attack.

7 Acknowledgements

All authors wish to acknowledge the support of the eCrypt-2 Network of Excellence funded by the Framework 7 programme of the European Union. The first author was partially funded by EPSRC and Trend Micro. The third author was also supported by a Royal Society Wolfson Merit Award.

References

1. S.S. Al-Riyami and K.G. Patterson. Certificateless public key cryptography. In *Advances in Cryptology – ASIACRYPT 2003*, Springer LNCS 2894, 452–473, 2003.

2. F. Armknecht, A. Peter and S. Katzenbeisser. A cleaner view on IND-CCA1 secure homomorphic encryption using SOAP. IACR e-print 2010/501, `http://eprint.iacr.org/2010/501`, 2010.
3. J. Baek, R. Steinfeld and Y. Zheng. Formal proofs for the security of signcryption. *Journal of Cryptology*, **20(2)**, 203–235, 2007.
4. M. Bellare and A. Palacio. Towards Plaintext-Aware Public-Key Encryption without Random Oracles. In *Advances in Cryptology – ASIACRYPT 2004*, Springer LNCS 3329, 37-52, 2004.
5. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption. In *Advances in Cryptology – EUROCRYPT'94*, Springer LNCS 950, 92-111, 1994.
6. D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1 In *Advances in Cryptology – CRYPTO '98*, Springer LNCS 1462, 1–12,1998.
7. R. Cramer, R. Gennaro and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology – EUROCRYPT '97*, Springer LNCS 1233, 103–118, 1997.
8. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology – CRYPTO '98*, Springer LNCS 1462, 13–25, 1998.
9. I. Damgård Towards practical public-key schemes secure against chosen ciphertext attacks. In*Advances in Cryptology – CRYPTO '91*, Springer LNCS 576, 1991.
10. I. Damgård, J. Groth and G. Salomonsen. The theory and implementation of an electronic voting system. In *Secure Electronic Voting*, Kluwer Academic Publishers, 77–99, 2002.
11. A. Dent. A designer's guide to KEMs. In *Coding and Cryptography 2003*, Springer LNCS 2898, 133–151, 2003.
12. M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in Cryptology – EUROCRYPT 2010*, Springer LNCS 6110, 24–43, 2010.
13. C. Gentry. Fully homomorphic encryption using ideal lattices. In *Symposium on Theory of Computing – STOC 2009*, ACM, 169–178, 2009.
14. C. Gentry. A fully homomorphic encryption scheme. Manuscript, 2009.
15. C. Gentry and S. Halevi. Implementing Gentry's fully-homomorphic encryption scheme. Manuscript, 2010
16. Z.-Y. Hu, F.-C. Sun and J.-C. Jiang. Ciphertext verification security of symmetric encryption schemes. *Science in China Series F*, **52(9)**, 1617–1631, 2009.
17. M. Joye, J. Quisquater, and M. Yung. On the power of misbehaving adversaries and security analysis of the original EPOC. In *Topics in Cryptography – CT-RSA 2001*, Springer LNCS 2020, 208–222, 2001.
18. H. Lipmaa. On the CCA1-security of ElGamal and Damgård's ElGamal. To appear *Information Security and Cryptology – INSCRYPT 2010*, 2010.
19. J. Manger. A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as standardized in PKCS # 1 v2.0 In *Advances in Cryptology – CRYPTO '01*, Springer LNCS 2139, 230–238, 2001.
20. R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, 169–177, 1978.
21. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal ACM*, **56(6)**, 1–40, 2009.
22. N.P. Smart. Breaking RSA-based PIN encryption with thirty ciphertext validity queries. In *Topics in Cryptology – CT-RSA 2010*, Springer LNCS 5985, 15-25, 2010.

23. N.P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography – PKC 2010*, Springer LNCS 6056, 420–443, 2010.