On FHE Without Bootstrapping
(Informal)

Aayush Jain

Indian Institute of Technology,
Delhi, India
aayushjainiitd@gmail.com

Abstract. In this work we come up with two fully homomorphic schemes. First, we propose an IND-CPA secure symmetric key homomorphic encryption scheme using multivariate polynomial ring over finite fields. This scheme gives a method of constructing a CPA secure homomorphic encryption scheme from another symmetric deterministic CPA secure scheme. We base the security of the scheme on pseudo random functions and prove the scheme to be IND-CPA secure, rather than basing security on hard problems like Ideal Membership and Gröbner basis as seen in most polly cracker based schemes which also use multivariate polynomial rings. This scheme is not compact but has many interesting properties- It can evaluate circuits of arbitrary depths without bootstrapping for bounded length input to the algorithm. Second, we also describe another similar symmetric key scheme which is compact, fully homomorphic and doesn't require bootstrapping. The scheme is on the lines of the work of Albrecht et. al. (Asiacrypt-2011) and is proven to be bounded CPA secure. Proof is based on Ideal Membership/ Ideal Remainder/Gröbner basis problem.

Keywords: Fully Homomorphic Encryption, Multivariate Polynomials, Bootstrapping, Symmetric Key Cryptography

1 Introduction

There have been schemes based on Gentry's blueprint like the [1], [2] scheme. Problem with those is inefficient bootstrapping and huge keys and cipher text sizes. [3] tells us that it is possible to create a public FHE from a symmetric key FHE. We have also seen a construction of public key homomorphic crypto system from a symmetric key crypto system in DGHV paper [2]. Hence, we will just consider symmetric key cryptosystems here. Let us examine, what goes wrong with having an FHE?

Consider the DGHV scheme which is probably simplest to understand: Secret is an odd number p.
$KeyGen(\lambda)$: Output a secret odd number p depending on security parameter.
Plaintext space : $\{0, 1\}$
$Encrypt(p, b)$: Output $p \times q + 2 \times r + b$, where q is a random number and r is a low norm random number depending

on λ

$Decrypt(p, c)$: Output (c mod p) mod 2

Why is it Somewhat homomorphic? Because, If a cipher text has the form $p \times Q + 2 \times R + B$ where B is the bit in the plaintext space to be encrypted, the decryption algorithm outputs B correctly as long as $|2 \times R + B| \leq p$. When we multiply cipher-texts (or add many of them) $|2 \times R + B|$ part grows and becomes more than p so the decryption algorithm outputs $B' \neq B$ where $p \times Q + 2 \times R + B = p \times Q' + 2 \times R' + B'$ and $|2 \times R' + B'| \leq p$. This is what happens in scheme's based on Gentry's blue print using ideal lattices.

One solution: What if we encrypt b now in $[0, p-1]$ as follows:

$KeyGen$: Same as before as in the DGHV scheme.

$Encrypt(p, b)$ output $p \times q + b$ for a random q.

$Decrypt(p, c)$ Output c mod p.

This scheme would be homomorphic and work fine but would no longer be secure! This is because if an eavesdroper has two encryptions of 0 : $p \times q_1$ and $p \times q_2$, and he takes gcd of those, he would recover p. Now in this(insecure) scheme observe that $Encrypt(b)$ outputs $b + i$ where $i \in I = (p)$ in $\mathbb{Z}$. For such a scheme to be secure we atleast want that the ideal I should have (practically)infinite or exponential number of generators. Every Ideal in $\mathbb{Z}$ in principal. Number rings have ideals that are generated with 2 generators. We will have to look at rings that have ideals that have large number of generators. For this project we propose analysing the ring of multivariate polynomials $F_q[t_1, t_2, ..., t_N]$ where F_q is a finite field.

2 Related Work

After Gentry's initial kick to the field of homomorphic encryption whole new ideas have emerged in a short span of time. Majority of work has been done on lattice based primitives. Gentry based his scheme on ideal lattices. [2] presents a simple construction using integers and explored the fact that a public key homomorphic encryption can be built based on a secret key scheme. [5] presents a scheme based on the LWE problem by Brakerski and Vaikunthanathan. Gentry and Halevi, have been able to implement all aspects of Gentry's scheme in [6] including the bootstrapping step. This work was an improvement to [15]. Bootstrapping step renders the scheme impractical and hence recent constructions lie [7], [8] aim to avoid it.

We base our scheme on rings of multivariate polynomials and there has been a lot of work in this area. [4] is the main reference to this work. This paper generalizes our second scheme to a generic construction. Bounded CPA security of our second scheme follows directly from [4]. Schemes outlined in [4] is based upon Gröbner basis/ Ideal Remainder/Ideal Membership problem. Any of these problem reduce to any other of these. We will not be delving into these problems and the security proof for the second scheme and for detailed treatment refer [4].

In 1993, Barkee et al. wrote a paper [9] to challenged that one should not base crypto on Gröbner basis theory. This was done by proposing a scheme and highlighting a fact that it can be broken in singly exponential time using ideas in [12]. Subsequently, there have been many proposals. All of them were broken by attacks. [11] gives a very good survey of polly cracker style schemes and attacks. The only scheme that is not broken is [10], which is closely related to lattices.

3 Our Contributions

This leads us to the motivation of this work: What goes wrong in having a FHE without bootstrapping? Intutively, When one multiplies(or adds) cipher-text, the size of the cipher-text grows. In order to fix that we go for ”noise” based schemes on lattices. Introducing noise makes the scheme somewhat homomorphic and one has to come up with bootstrapping and squashing etc. to make it fully homomorphic. When one tries to design a homomorphic scheme without using ”noise”, compactness and security becomes a problem. Schemes using ”noise” are based on established hard problems like the LWE, Approximate GCD problems etc. while those without noise are based on problems like Gröbner basis problem and the ideal membership problems whose average case hardness is not known. Compactness is ensured by publishing set of encryptions of objects depending on the secret key, though this is not always possible. A similar thing is done in the second scheme we describe in this paper. In noisy schemes we output a similar set for bootstrapping, but we have to typically squash the decryption circuit to a lower depth and this new scheme leads to even more huge cipher-text. Our second construction avoids this.
Currently, most homomorphic Encryption scheme are impractical and characteristic of the following issues:

- Bootstrapping
- Squashing step
- Huge cipher text

In this work we propose two scheme. First, we come up with a CPA secure, symmetric key, non compact, fully homomorphic scheme that can’t be made into a public key scheme using the known standard transformations. Scheme uses for its construction a randomly chosen member of a family pseudo-random functions and has some very interesting properties. Second, we propose a scheme that is bounded CPA secure, symmetric key, fully homomorphic and doesn’t require squashing. Since it is bounded CPA secure it can’t be made into a public key scheme. This scheme is based on symmetric polly cracker scheme from [4]. Their scheme is not compact and we come up with a transformation to make it compact.

4 Preliminaries

In this paper, wherever we have a set S, $s \xleftarrow{\$} S$ denotes sampling of an element $s \in S$ randomly and uniformly(unless specified).

4.1 Homomorphic Encryption

In this work we consider symmetric key homomorphic encryption with respect to the addition and multiplication gates in the ring form by plain-text space. A homomorphic encryption scheme ε has four algorithms: the usual *KeyGen*, *Encrypt*, and *Decrypt*, and an additional algorithm *Evaluate*. The algorithm *Evaluate* takes as input a a circuit $\mathcal{C}$, a tuple of ciphertexts $\boldsymbol{c} = (c_1, ..., c_t)$ (one for every input of $\mathcal{C}$), and outputs another ciphertext c using publicly available information(typically some function of the secret key).

Definition 1. *(Correct Homomorphic Decryption).*
The scheme $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$ is correct for a given t-input circuit $\mathcal{C}$ if, for any key sk output by $KeyGen(\lambda)$, any t plaintexts $m_1, .., m_t$, and any cipher-texts $\boldsymbol{c} = c_1, ..., c_t$ with $c_i \leftarrow Encrypt_{\varepsilon}(sk, m_i)$, it is the case that: $Decrypt(sk, Evaluate(\mathcal{C}, \boldsymbol{c})) = \mathcal{C}(m_1, ..., m_t)$

Definition 2. *(Homomorphic Decryption).*
The scheme $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$ is homomorphic for a class $\mathcal{C}$ of circuits if it is correct $\forall$ circuits $C \in \mathcal{C}$. ε is fully homomorphic if it is correct for all boolean circuits.

The semantic security(IND-CPA) of a homomorphic encryption scheme is defined in the usual way [14], without reference to the *Evaluate* algorithm. (Indeed *Evaluate* is a public algorithm with no secrets.)

The "real challenge" in constructing fully homomorphic encryption comes from the compactness property, which essentially means that the size of the cipher-text that Evaluate generates does not depend on the size of the circuit C.

Definition 3. *(Compact Homomorphic Encryption).*
The scheme $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$ is compact if there exists a fixed polynomial bound $b(\lambda)$ so that for any key sk output by $KeyGen(\lambda)$, any circuit $\mathcal{C}$ and any sequence of cipher-text $\boldsymbol{c} = (c_1, ..., c_t)$ that was generated with respect to sk, the size of the cipher-text $Evaluate(\mathcal{C}, \boldsymbol{c})$ is not more than $b(\lambda)$ bits (independently of the size of $\mathcal{C}$)

If a scheme can evaluate class of circuits with bounded-depth correctly it is called Somewhat homomorphic.

Definition 4. *(Augmented Decryption Circuits).*
Let $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$ be an encryption scheme, where decryption is implemented by a circuit that depends only on the security parameter. For a given value of the security parameter λ, the set of augmented decryption circuits consists of two circuits, both take as input a secret key and two ciphertexts: One circuit decrypts both ciphertexts and adds the resulting plaintext, the other decrypts both ciphertexts and multiplies the resulting plaintext bits. We denote this set by $D_{\varepsilon}(\lambda)$

Definition 5. *(Bootstrappable Encryption).*
Let $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$ be a homomorphic encryption scheme, and for every value of the security parameter λ let $\mathcal{C}_\varepsilon(\lambda)$ be a set of circuits with respect to which ε is correct. We say that ε is bootstrappable if $D_\varepsilon(\lambda) \subseteq \mathcal{C}_\varepsilon(\lambda)$ holds for every λ.

Now, [1] says that given a bootstrappable somewhat homomorphic encryption scheme it is possible to construct a compact and secure leveled homomorphic encryption (one that can evaluate circuits of depth d for an input d). If the scheme ε is "KDM" or "circular secure" then its possible to make this scheme fully homomorphic using explicit transformations- a process called *Bootstrapping*. Since both the schemes we present is fully homomorphic inherently, we don't talk about bootstrapping.

4.2 Fundamentals of Gröbner basis Theory

We refer to [4] for detailed Gröbner theory and list out the main points required. Assume the ring $P = F_q[t_1, .., t_N]$ the ring of multivariate polynomials over the finite field F_q having q elements. Assume q to be prime. We consider a polynomial ring P, some monomial ordering on elements of P. We denote by $M(f)$ the set of all monomials appearing in $f \in P$. By $LM(f)$ we denote the leading monomial appearing in $f \in P$ according to the chosen term ordering. We denote by $LC(f)$ the coefficient $\in F_q$ corresponding to $LM(f)$ in f and set $LT(f) = LC(f)LM(f)$. We denote by $P_{<d}$ the set of polynomials of degree $< d$ (and analogously for $>, \leq, \geq$, and $=$ operations).We define $P_{=0}$ as the underlying field including $0 \in F_q$. We define $P_{<0}$ as zero. Finally, we denote by $M_{<m}$ the set of all monomials $< m$ for some monomial m (and analogously for $>, \leq, \geq$, and $=$ operations).We assume the usual power product representation for elements of P.

Definition 6. *(Gröbner basis). Let I be an ideal of $P = F[x_1, ..., x_{n-1}]$ and fix a monomial ordering. A finite subset $G = \{g_0, ..., g_{m-1}\} \subset I$ is said to be a Gröbner basis of I if for any $f \in I$ there exists $g_i \in G$ such that $LM(g_i) \mid LM(f)$.*

For a set of polynomials S denote $V(S)$ as the set of common zeros or algebraic set corresponding to S. In this paper we refer to this set as variety of S although it is an abuse of notation.

It is possible to extend the division algorithm to multivariate polynomials: we write $r = f \bmod G$ when $f = \Sigma_{i=0}^{i=n-1} h_i g_i + r$ with $M(r) \cap < LM(G) >= 0$. When G is a Gröbner basis r is unique and is called the normal form of f with respect to the ideal I. In particular we have that $f \bmod I = f \bmod G = 0$ if and only if $f \in I$. Together P and I define the quotient ring P/I and, by abuse of notation, we write $f \in P/I$ if $f \bmod I = f$ where equality is interpreted as those on elements of P. That is, we identify elements of the quotient P/I with their minimal representation in P.

Definition 7. *(Reduced Gröbner basis). A reduced Gröbner basis for an ideal $I \subset P$ is a Gröbner basis G such that:*

1. *$LC(g) = 1\ \forall g \in G$*
2. *$\forall g \in G$, $\nexists m \in M(g)$ such that m is divisible by some element of $LM(G \diagdown \{g\})$*

[4] provides an algorithm *ReduceGB*(G) to find a reduced Gröbner basis from a Gröbner basis and it is unique i.e. Reduced gröbner basis for an ideal is unique and we refer to the paper for the algorithm. As outlined in [4], Buchberger's Criterion provides a criterion to check if a set forms a Gröbner basis, using "S polynomials". We now state an important result, refer [4] for a proof.

Theorem 1. *A set $\{g_1, .., g_N\} \subset P$ with $LM(g_i) = t_i^{d_i}$ with $d_i \geq 0 \forall i \in [1, N]$ is a Gröbner basis.*

This theorem motivates us algorithms to construct Gröbner basis. We are interested in Gröbner basis with a non-empty variety i.e. for $P = F_q[t_1, .., t_N]$ we will be interested in Gröbner basis $G = \{g_0, .., g_{N-1}\}$ such that $V(G) \neq \phi$(equivalently $\exists v \in F_q^N$ such that $g_i(v) = 0 \forall i \in [1, N]$). Hilbert Nullstellensatz for finite fields say the following [16]:

Theorem 2. *For an arbitrary Finite Field F_q, given m polynomials $f_1, .., f_m \in F_q[t_1, .., t_N]$ have no common zero in F_q^N if and only if $1 \in < f_1, .., f_m, t_1^q - t_1, .., t_N^q - t_N > \subseteq F_q[t_1, .., t_N]$*

Hence with a good chance an ideal will have a non-empty variety(or algebraic set). Let's discuss algorithms now:

GBGen(λ, P, d): Generates a Gröbner basis for an ideal in P with a non-empty variety where the generators have a degree d. In actual instantiation we would replace this with $GBGen_{dense}(\lambda, P, d)$ which does the following.

Algorithm 1 $GBGen_{dense}(\lambda, P, d)$

```
$(a_1, .., a_N) \leftarrow (F_q*)^N$
for $i \in [1, N]$ do
    $g_i \leftarrow t_i^d$
    for $m_j \in M_{<t_i^d}$ do
        $c_{i,j} \xleftarrow{\$} F_q$
        $g_i \leftarrow g_i + c_{i,j} m_j$
    end for
    $g_i \leftarrow g_i - g_i(a_1, .., a_N)$(still forms a GB)
end for
$G \leftarrow \{g_1, .., g_N\}$
return ReduceGB$(G), a_1, .., a_N$
```

So based on this property of Gröbner's basis three problems have been defined and here we will give an informal definition of all these problems. For details refer [4]. Assume there is an oracle $\mathcal{O}$ which takes as input ring P of polynomials over $n(\lambda)$(is a polynomial) finite field F_q of characteristic $q(\lambda)$, a Gröbner basis G having n elements, a constant d which is the degree of the elements in G, a constant b which is the maximum degree of polynomial released by the oracle. Assume $d < b$. $\mathcal{O}$ returns random polynomials of degree at most b in the Ideal generated by G. The problems also depends on apriori fixed polynomial $m()$ which denotes the maximum number of queries made to $\mathcal{O}$.

- **Gröbner Basis Problem(GB)**: The game is as follows. Challenger samples a ring P and G. It then gives adversary $\mathcal{A}$ an access to $\mathcal{O}$ which can be queried at most $m()$ times. Adversary has to output a reduced Gröbner basis of Ideal generated by G. $\mathcal{A}$ wins if it returns a correct Gröbner basis.
- **Ideal Remainder Problem(IR)**: The game is as follows. Challenger samples a ring P and G. It then gives adversary $\mathcal{A}$ an access to $\mathcal{O}$ which can queried at most $m()$ times. Challanger than challenges $\mathcal{A}$ with a random polynomial in $P_{\leq b}$, f. Adversary has to output $r \leftarrow f \bmod G$. $\mathcal{A}$ wins if it answers correctly.
- **Ideal Membership Problem(IM)**: The game is as follows. Challenger samples a ring P and G. It then gives adversary $\mathcal{A}$ an access to $\mathcal{O}$ which can be queried at most $m()$ times. Challanger than challenges $\mathcal{A}$ either with a random polynomial in $P_{\leq b}$ or a member of the ideal generated by G, f. Adversary has to output if f is in ideal generated by G or not. $\mathcal{A}$ wins if it answers correctly.

[4] gives a reduction of each of these problems to each other when $q^{dimP/<G>}$ is small i.e. polynomial in λ. If we assume any one to be hard then other two are equally hard. More generally $GB \geq IR \geq IM$.

Paper [4] also suggests that it is reasonable to assume:

Definition 8. *(GB/IR/IM Assumption).*
Let P be such that $n(\lambda) = \Omega(\lambda)$. Assume $b - d > 0$, $b > 1$, and that $m(\lambda) = cn(\lambda)$ for a constant $c \geq 1$. Then the advantage of any ppt algorithm in solving the $GB/IR/IM$ problem is negligible as function of λ.

4.3 Pseudo-Random Functions(PRF)

Let $\mathcal{G} : \mathcal{K} \times \{0,1\}^{\lambda} \rightarrow \{0,1\}$. be a family of keyed function where $\mathcal{K}$ is the key space.

Let $\mathcal{A}$ be an adversary. A uniform random key K is chosen from $\mathcal{K}$ and $\mathcal{A}$ is given access to the oracle $g_K \in \mathcal{G}$. $\mathcal{A}$ can make queries to $g_K()$ and receives the responses. At the end of the interaction, $\mathcal{A}$ outputs a bit b. Denote by $\mathcal{A}^{g_K} \Rightarrow 1$ the event that $\mathcal{A}$ outputs the bit 1 after its interaction with the oracle g_K.

Let $g*$ be a function chosen uniformly at random from the set of all functions which map $\{0,1\}^{\lambda}$ to $\{0,1\}$. Consider now the interaction of A with $g*$ instead of g_K. Denote $\mathcal{A}^{g*} \Rightarrow 1$ to be the event that A outputs 1 after its interaction with the oracle $g*$.

The advantage of $\mathcal{A}$ in breaking the PRF property of g_K is defined to be

$Prob[A^{g_K} \Rightarrow 1] - Prob[A^{g*} \Rightarrow 1]$.

In concrete terms we wish this advantage to be small. (This can be formalised in an asymptotic sense by requiring $\mathcal{A}$ to be PPT in the security parameter and the advantage to be negligible in the security parameter.)

5 Our first Construction

Fix the ring as $P = F_q[t_1, t_2, ..., t_N]/(t_1^q - t_1, .., t_N^q - t_N)$. F_q is a finite field with q elements. q is chosen to be $O(1)$. For this work we choose $q = 2$. Analysis is similar for fields of higher characteristic. Idea we propose is to have N is exponential in the security parameter or $\Omega(2^\lambda)$. As discussed earlier we would only be describing a symmetric key crypto system.

Let there be two parties, Alice and Bob. We want them to have as a shared secret a secure function $g : [1, N] \to F_q$. g is sampled randomly from the family of pseudo random functions $\mathcal{G} = \{g_k \mid K \in \mathcal{K}\}$. A secret key K is chosen uniformly at random from $\mathcal{K}$. The randomness in g arises solely from the randomness in the choice of K. A simple way to instantiate g using a block cipher is as follows: Let E_K be a block cipher which maps λ-bit strings to λ-bit strings. Define $g_K(X)$ to be the first bit of E_K, i.e., apply E_K to the λ-bit string X and take only the first bit. (There is nothing special about the first bit, and other bit would also do). This is quite a simple construction. Let's us describe our first candidate scheme $\pi = (KeyGen, Enc, Dec, Eval)$ now.

KeyGen(λ): Output a secret function g sampled randomly from the family of pseudo random functions $\mathcal{G} = \{g_k \mid K \in \mathcal{K}\}$. $\mathcal{G} : \mathcal{K} \times [1, N] \to F_2$. g is shared secret. Alternately, we could have stored as secret key a vector which stores a sequence in the field F_q, $(a_1, ..., a_N)$. Since N is exponential in the security parameter, it would make the *KeyGen* scheme exponential in time and space. This is the reason we just store a secret function and $g(n)$ is calculated whenever required. Ideal in the ring that we will be using is $I = (t_1 - g(1), ..., t_N - g(N))$. This is the set $I = \{\sum_{i=0}^{i=N} (t_i - g(i)) \times f_i(t_1, ..., t_N)\}$ where $f_i(t_1, ..., t_N)$ are random polynomials in the ring. Our plain-text space is $\mathcal{P} = F_2$

Theorem 3. *Let P be the ring described above, I be the ideal $(t_1 - g(1), .., t_N - g(N))$ then, $i(t_1, .., t_N) \in I$ iff $i(g(1), .., g(N)) = 0$ [13]*

Encrypt(g, b): Select m numbers from $[1, N]$(can be repeated). m is $O(log(\lambda))$ for reasons described later. Order them in ascending order, denote this random number vector $\boldsymbol{n} = (n_1, n_2, ..., n_m)$. To encrypt a bit b, do the following: $f \xleftarrow{\$} F_2[t_{n_1}, .., t_{n_m}]/(t_{n_1}^2 - t_{n_1}, .., t_{n_m}^2 - t_{n_m})$. This f is linear in m selected indeterminates.
Output $c \leftarrow b + f(t_{n_1}, .., t_{n_m}) - f(g(n_1), .., g(n_m))$. Alternately, for each encryption choose a single $log_2(N)$-bit integer n and set $n_i = n + i - 1 \bmod N \forall i \in [1, m]$. In both cases, the output bits of $g()$ will appear to be independent and uniformly distributed to a computationally bounded adversary.

Note that cipher text is a polynomial with at most m variables. Since, we are working an extension ring of F_2, for our purpose, $t_i^2 = t_i \forall i \in [1, N]$ when evaluated at 0 and 1. The multiplication is done using the rule

$t_i^2 = t_i \forall i \in [1, N]$. For example, $(t_1t_5t_{11} + t_2) \times (t_7 + t_5) = t_1t_5t_{11}t_7 + t_2t_7 + t_1t_5t_{11} + t_2t_5$. This makes cipher-text is linear in all indeterminates.

Decrypt$(g, c(t_1, ..., t_N))$: Evaluate the cipher-text polynomial at $t_i = g(i) \forall i \in [1, N]$, as usual. Formally,
Output, $b \leftarrow c(g(1), .., g(N))$

This is a polynomial time algorithm because the cipher text polynomial is linear in all m indeterminates, such a cipher text can have 2^m monomials. Since m is chosen to be $O(log(\lambda))$ the length of the cipher-text is at most a polynomial in the security parameter. Since each cipher text is a function of at most m variables, decryption algorithm computes $g(*)$ on at most $O(log(\lambda))$ points and evaluates a polynomially long cipher-text, decryption takes polynomial number of operations.

Evaluate$(c_1(t_1, ..., t_N), c_2(t_1, ..., t_N))$: It would be sufficient to describe Addition and Multiplication gates for the purpose of describing *Evaluate* algorithm. Let's define add (similarly multiplication- replace + with × in the argument) in the following manner:

Add$(c_1(t_1, ..., t_N), c_2(t_1, ..., t_N))$: first compute
$c(t_1, ..., t_N) = c_1(t_1, ..., t_N) + c_2(t_1, ..., t_N)$.
$c(t_1, ..., t_N)$ will be a polynomial in at most $2m$ variables if the input cipher-text is fresh. Multiplication is done similarly,

In summary,

Add$(c_1(t_1, ..., t_N), c_2(t_1, ..., t_N))$:
Compute $c(t_1, ..., t_N) = c_1(t_1, ..., t_N) + c_2(t_1, ..., t_N)$

Mult$(c_1(t_1, ..., t_N), c_2(t_1, ..., t_N))$:
Compute $c(t_1, ..., t_N) = c_1(t_1, ..., t_N) \times c_2(t_1, ..., t_N)$

CORRECTNESS: Scheme is correct as encrypt algorithm takes as input a secret function g and a bit b in the plain text space F_2 and outputs an element in the coset $b + I$, where I is the ideal $(t_1 - g(1), .., t_N - g(N))$. Suppose *Encrypt*$(b, g)$ outputs $i(t_1, .., t_N) + b$ for $i(t_1, .., t_N) \in I$, *Decrypt* evaluates this cipher-text at $(g(1), .., g(N))$ and outputs $b + i(g(1), .., g(N)) = b$ since $i(t_1, .., t_N) \in I$ and by theorem 1 $i(g(1), .., g(N)) = 0$. Multiplication and addition works correctly because of the ring structure of the cipher texts.
It is observed when we keep on adding or multiplying various cipher-text the size of cipher-text grows and hence the scheme is not compact. For compactness it is desirable to have a procedure like cipher text reduction, which is based on the fact that intermediate cipher-text $c(t_1, .., t_N)$ depends upon at most $2m$ variables and would contain at most

2^{2m} monomials, which is polynomially bounded in the security parameter. Suppose that monomials appearing in the cipher-text look like $t_1^{e_1}t_2^{e_2}...t_N^{e_N}$ where exponents $e_i \in [0,1] \forall i \in [1,N]$ and at most $2m$ of the exponents are non zero. If we replace this monomial by $g(1)^{e_1}g(2)^{e_2}...g(N)^{e_N}+$ $Encrypt(g,0)$, we still get a valid cipher-text. Hence, we replace each monomial $t_1^{e_1}t_2^{e_2}...t_N^{e_N}$ with $g(1)^{e_1}g(2)^{e_2}...g(N)^{e_N}+$ $Encrypt(g,0)$, where the encryptions of 0 depend on selected m variables. This gives us a cipher-text depending upon at most m variables. One can check this is a polynomial time algorithm. Since, an untrusted sever cannot store exponential number of encryptions of product of the secret key's this is not how we achieve compactness. For this variant of scheme just consider add and multiply without any cipher-text reduction so that the cipher text size increases with multiplication and addition. In the next scheme, we describe *degreereduction* using similar concepts which ensures compactness. Since we want our cipher text size to be polynomially bounded we can only solve circuits of bounded depth.
Lets analyse depth using 2 fan in addition gates and 2 fan in multiplication gates. If we start with a fresh cipher-text having m variables, at a depth d we have about $2^d.m$ variables. At this depth the maximum length of cipher text is 2^{m2^d}. When we want length of the final cipher-text to be bounded by a polynomial $l(\lambda)$ in that case the depth that we can solve is $loglog(l(\lambda)) - log(m)$, which is a constant. But this is an overkill. If we somehow encrypt such that fresh-cipher text has at most $O(1)$ length, we can solve $O(log(log(\lambda)))$ deep circuits. This is because at each level the length of the cipher text squares so at level d it will be c^{2^d} and for this to be polynomially bounded we require d to be $O(loglog(\lambda))$, for a constant c.
When m is $\theta(log(\lambda))$ and given a cipher-text tuple having $O(1)$ fresh cipher-text we can evaluate circuits of all depths. When m is $O(1)$, and input cipher-text vector has $O(log(\lambda))$ cipher-texts, we can evaluate arbitrary deep circuits. This is because number of variables appearing in the cipher text vector is still $O(m)$, and hence it produces a bound on the length of the final cipher-text.

5.1 Proof of Security

For arguing about the security of the scheme we consider number of games. Denote $Func_N()$ as set of all functions from $[1,N] \rightarrow F_2$. $\mathcal{G}$ denote the set of pseudo-random functions from same domain and co-domain $\mathcal{G} = \{g_k \mid k \in \mathcal{K} = \{0,1\}^\lambda\}$. Consider these definition of games played between a challenger $\mathcal{C}$ and any ppt adversary $\mathcal{A}$.

Game 0:

Setup:

Challenger

$sk \leftarrow g_k \xleftarrow{\$} \mathcal{G}$. This is viewed as a secret key for encryption scheme π described above.

Query Phase 1:

This is the query phase where $\mathcal{A}$ queries challenger $\mathcal{C}$ encryptions of messages. $\mathcal{A}$ returns queries by using $Encrypt(sk,*)$.

Challenge Phase:

$\mathcal{C}$ picks $\beta \xleftarrow{\$} \{0,1\}$ and sends $Encrypt(sk,\beta)$ to $\mathcal{A}$.

Query Phase 2:

Same as Query Phase 1.

Guess

$\mathcal{A}$ tries to guess β and outputs β'. If $\beta = \beta'$, $\mathcal{A}$ wins the game.

Advantage of an adversary $\mathcal{A}$ in wining the the Game 0 is defined as $Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) = Pr[\beta = \beta'] - 1/2$.

Game 1:

Game 1 is played same as Game 0 except that in **setup** phase Challenger assigns secret key as $sk \leftarrow f \leftarrow^{\$} Func_N()$ $Adv_{\mathcal{A},\pi}^{Game-1}(\lambda)$ is defined analogously.

Note that Game 0 is the same as the IND-CPA game for the scheme π. So, $Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) = Adv_{\mathcal{A},\pi}^{IND-CPA}(\lambda)$. We will now prove that $Adv_{\mathcal{A},\pi}^{IND-CPA}(\lambda)$ is negligible for any ppt adversary $\mathcal{A}$.

Lemma 1. *If there is an Adversary $\mathcal{A}$ for which $\mid Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) - Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) \mid$ is not negligible then, we can construct a ppt adversary $\mathcal{D}$ which distinguishes between a function chosen randomly from a family of pseudo random functions $\mathcal{G}$ and a truly random function with a non-negligible advantage*

Proof. Let, $\mid Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) - Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) \mid= \varepsilon$. Let us consider an adversary $\mathcal{D}$ that uses $\mathcal{A}$. Consider $\mathcal{D}$ that receives a function h from the challenger and $\mathcal{D}$ uses $\mathcal{A}$ and plays IND-CPA game with A and answering queries by using h as the secret key. Suppose $\mathcal{D}$ challenges $\mathcal{A}$ by encrypting β and $\mathcal{A}$ outputs β' then $\mathcal{D}$ returns $xor(\beta, \beta')$. For $\mathcal{D}$ the problem is the same as to tell whether $\mathcal{A}$ is in game 0 or game 1. For this distinguisher, $\mathcal{D}$, PRF-Advantage is defined as $\mid Pr[f \leftarrow Func_N \mid D(f) = 1] - Pr[g_k \leftarrow \mathcal{G} \mid D(g_k) = 1] \mid$. This is also equal to $\mid Pr[f \leftarrow Func_N \mid D(f) = 0] - Pr[g_k \leftarrow \mathcal{G} \mid D(g_k) = 0] \mid$. Note that, $Pr[f \leftarrow Func_N \mid D(f) = 0] = 1/2 + Adv_{\mathcal{A},\pi}^{game-1}(\lambda)$ and $Pr[g_k \leftarrow \mathcal{G} \mid D(g_k) = 0] = 1/2 + Adv_{\mathcal{A},\pi}^{game-0}(\lambda)$. So, PRF-Advanage for $\mathcal{D}$ comes out to be, $\mid Adv_{\mathcal{A},\pi}^{game-0}(\lambda) - Adv_{\mathcal{A},\pi}^{game-1}(\lambda) \mid= \varepsilon$.

Lemma 2. $Adv_{\mathcal{A},\pi}^{game-0}(\lambda)$ *is negligible for any ppt adversary $\mathcal{A}$.*

Proof. Consider the following game,

Game 2

Setup:

Challenger

$sk \leftarrow f \leftarrow^{\$} Func_N()$. This is viewed as a secret key for encryption scheme π described above.

Query Phase 1:

This is the query phase where $\mathcal{A}$ queries challenger $\mathcal{C}$ encryptions of messages. $\mathcal{A}$ returns queries by using $Encrypt(sk, *)$.

Challenge Phase:

$\mathcal{C}$ picks $\beta \leftarrow^{\$} \{0, 1\}$ and sends $Encrypt(sk, \beta)$ to $\mathcal{A}$.

Query Phase 2:

Same as Query Phase 1.

Guess

If challenge cipher-text depends on the variable which has been encountered before in a queried cipher-text, abort the game call this event $\mathcal{F}$. Otherwise, $\mathcal{A}$ tries to guess β and outputs β'. If $\beta = \beta'$, $\mathcal{A}$ wins the game. $Adv_{\mathcal{A},\pi}^{Game-2}(\lambda) = Pr[\beta = \beta'] - 1/2$

Observe that Game-2 is similar to Game-1 except when challenge cipher-text depends on variables on which the queried cipher-text replies depend on. At this event $\mathcal{F}$, Game-2 aborts.

By using the difference lemma [17],

$\mid Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) - Adv_{\mathcal{A},\pi}^{Game-2}(\lambda) \mid \leq pr[\mathcal{F}]$.

$Adv_{\mathcal{A},\pi}^{Game-2}(\lambda) = 0$, because in this game challenge cipher-text depend upon completely new set of variables and decryption involves evaluation of this cipher-text on random set of points(output of a truly random function). Lets, calculate $Pr[\mathcal{F}]$. The probability that $\mathcal{F}'$ occurs is the probability that challenge cipher-text doesn't depend on variables involved in κ cipher-text queries replied.
$Pr[\mathcal{F}'] \geq (\frac{N-m\kappa}{N})^m \approx 1 - \frac{m^2\kappa}{N}$ as long as κ is $O(N)$ and grows slower than N. Since, N is exponential, κ is allowed to be exponential for this game. $Pr[\mathcal{F}] \leq \frac{m^2\kappa}{N}$. This implies that, $\mid Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) - Adv_{\mathcal{A},\pi}^{Game-2}(\lambda) \mid \leq \frac{m^2\kappa}{N}$ or $Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) \leq \frac{m^2\kappa}{N}$, which is negligible. Since, $\mid Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) - Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) \mid = \varepsilon$, $Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) \leq Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) + \varepsilon$, As long as κ is polynomially bounded, $\mathcal{D}$ makes polynomial queries with the function oracle and hence ε is negligible as PRF problem is hard. Combining theses results we get $Adv_{\mathcal{A},\pi}^{Game-0}(\lambda) \leq Adv_{\mathcal{A},\pi}^{Game-1}(\lambda) + \varepsilon$ is negligible.

5.2 What we achieve from the scheme

As one can clearly see, fresh cipher text polynomials are polynomials in at most m variables and each cipher text can have at most 2^m monomial terms. Since the cipher-text size grows, the scheme is not compact, as seen in polycracker based scheme such as [4].

For an input cipher-text vector of upto a constant length, the scheme can evaluate circuits of all depths. We worked on field with characteristic 2 but one could generalize the discussion above with field of higher characteristic q as long its length is bounded by a constant. We base the CPA security of the scheme on PRF's rather than basing them on problems like Gröbner basis/ Ideal Membership/ Ideal Remainder which yield us at most bounded security.
Implementations [6], [15] show that homomorphic systems have huge cipher-texts and generally cipher length is bounded by large degree polynomials in λ. In our scheme, cipher-texts can be very small, as small as $O(\lambda)$ without compromising on the security. This is achieved when cipher-text depends on $O(1)$ number of variables. This gives us an efficient cryptosystem for circuits of smaller depths.
Evaluate algorithm can solve circuits of every depth when the input cipher-text vector has length $O(log(\lambda))$ variables. This means it can handle for any depth, constant number of cipher-text having $O(log(\lambda))$ variables or $O(log(\lambda))$ cipher-texts with constant number of variables.

This also gives us a way of constructing a CPA secure symmetric homomorphic encryption scheme from a deterministic CPA secure symmetric encryption scheme which can be used as g function described in the scheme.

5.3 Achieving Information Theoretic Security

Consider a variant of the scheme where in the keys are generated on the fly! Suppose Alice and Bob wants to communicate. Also assume that Alice and Bob has a secure channel of communication(achieved by using some other encryption scheme). Main idea for achieving information theoretic security is to replace $g()$ in the Encryption scheme by a truly random function. Note that when one encrypts a bit he just uses m values of the secret function and it is wasteful to store N random values. To fix this we could generate key at the time of encryption itself! These "function values" are stored and later used at the time of decryption/encryption. When Alice creates a value of the secret function it sends it to Bob using the secure channel and vice-versa. When one does that, the secret function mimics a truly random function. We denote the database for keys by G We now describe the scheme.

Encrypt(b): Pick m random numbers in $[1, N]$. Denote them by $n_1, .., n_m$. If $g(n_i) \forall i \in [1, m]$ doesn't exists in G then set $g(n_i) \xleftarrow{\$} \{0, 1\}$ in G and send the $g(n_i)$ to the other party(if it exists). Now randomly choose $f(t_{n_1}, .., t_{n_m}) \in P$ such that f depends on at most m variables. Set $c \leftarrow f - f(g(n_1), .., g(n_m)) + b$ as the cipher text.

Decrypt(c): Output $b \leftarrow c(g(1), .., g(N))$ using database G.
Evaluate : Same as the original scheme.

PROOF OF SECURITY: Proof of security follows from the fact that the Advantage of any adversary can make $O(N)$ queries in breaking Game-1 is negligible(refer security proof of the original scheme).

6 Second construction - N is $O(poly(\lambda))$

Scheme described above is secure for an exponential N. Once N is made to be $\Omega(poly(\lambda))$ we can make a similar scheme compact. We prove the scheme to be bounded CPA secure. Security based on Ideal membership problem follows from [4]. Since we choose q to be $\Omega(2^\lambda)$ we do not know of a reduction of Ideal Remainder(IR) problem to Ideal Membership(IM) problem.

6.1 Scheme

We are dealing with the ring $P = F_q[t_1, .., t_N]$ where N is a polynomial in λ. Here, q is taken to be prime characteristic of the field. Assume that $q = q(\lambda)$ is $\Omega(2^\lambda)$. The monomials in this ring are of the form $t_1^{e_1}...t_N^{e_N}$, where exponents take values over non-negative integers. Degree of a monomial is defined as $e_1 + ... + e_N$. Degree of a polynomial in this ring is the degree of the monomial occurring in the polynomial that has the maximum degree.

We describe our symmetric key scheme now. Ideas used here are the similar to the ideas described in the symmetric polly cracker scheme described in [4]. We first briefly describe the scheme. Suppose Alice and Bob wants to communicate. They share as secret key some "special" polynomials that generate an ideal $I = (f_1, .., f_N)$. These functions form a Gröbner basis for the ideal. It is assumed that $V(I) \neq \phi$. So, that $(a_1, .., a_N) \in V(I)$. This point forms the secret key along with the f_i's. Plaintext space $\mathcal{P}$ is F_q. To encrypt a message $\pi \in F_q$, output $\pi + i(t_1, .., t_N)$ such that $i(t_1, .., t_N) \in I$. Decryption amounts to evaluating the cipher text at any point in $V(I)$. This scheme is homomorphic as $\pi_1 + i_1 + \pi_2 + i_2 \in \pi_1 + \pi_2 + I$ and $(\pi_1 + i_1).(\pi_2 + i_2) \in \pi_1.\pi_2 + I \forall i_1, i_2 \in I \& \pi_1, \pi_2 \in \mathcal{P}$. Let's describe these algorithms in detail. Let us denote the scheme as $\varepsilon = (KeyGen, Encrypt, Decrypt, Evaluate)$

Assume there is an algorithm GBGen() that returns a Grobner basis with one point in its variety or set of zeros.

KeyGen(λ): Fix N as a polynomial in λ. From the security proof, we can release $\Omega(N)$ message-encryption pairs before security breaks down. Fix d a constant, the degree of the generators of the secret ideal and a constant b which is the degree of fresh cipher text.
$(f_1, .., f_N, a_1, .., a_N) \xleftarrow{\$} GBGen(\lambda, P, d)$
$f_i(a_1, .., a_N) = 0 \forall i \in [1, N]$. Public Parameters also include the set K which is by definition, $K = \{Encrypt(sk, a_1^{e_1}..a_N^{e_N}) \mid \sum_{i=0}^{i=N} e_i \in [1, b+1]\}$. K contains $O(N^b)$ encryptions and hence it is a polynomial sized set.

In practice, instead of using $GbGen()$ one could use $GbGen_{dense}()$ for $KeyGen()$. We will now describe $Encrypt(, *)$.
$Encrypt(sk, \pi)$ Select $f \xleftarrow{\$} P_{\leq b}$
$f \leftarrow f - f \bmod G$, G is the Gröbner basis of the secret ideal
Output $c \leftarrow f + \pi$
Note that size of the cipher-text is polynomially bounded as number of monomials of degree at most b is $O(N^b)$.

$Decrypt(sk, c)$ Output $\pi \leftarrow c(a_1, .., a_N)$.
Decryption is correct as a valid cipher-text is of the form $\pi + \sum_{i=1}^{i=N} h_i f_i$, evaluation of which(at $(a_1, .., a_N)$) gives π.

$Evaluate(K, C, \boldsymbol{c})$: This algorithm takes as input a cipher-text vector $\boldsymbol{c}$ a circuit C and a set K which has encryptions of all the product terms of $b + 1$ values of g functions. It replaces add and mul gates in the circuit with the following description:

$Add(c_1(t_1, .., t_N), c_2(t_1, .., t_N))$ compute $c(t_1, .., t_N) = c_1(t_1, .., t_N) + c_2(t_1, .., t_N)$. Since degree of $c(t_1, .., t_N)$ is less than or equal to b, output $c(t_1, .., t_N)$.

$Mul(c_1(t_1, .., t_N), c_2(t_1, .., t_N))$ compute $c(t_1, .., t_N) = c_1(t_1, .., t_N).c_2(t_1, .., t_N)$ where "." is multiplication in the ring. Since the degree of the cipher text increases, perform degree reduction(described) on the cipher-text.

$degreereduction(K, c(t_1, .., t_N))$: This algorithm takes as input the set K and the cipher-text polynomial whose degree has to be reduced. If we encounter a monomial of degree greater than or equal to $b+1$ say $t_1^{e_1}..t_N^{e_N}$ of degree at most $2b$, replace this with an encryption of $a_1^{e_1}..a_N^{e_N}$. This is done inductively using the set K. If we have a monomial of degree $b+1$ replace it by corresponding encryption from set K which is a polynomial of degree b. Otherwise for every monomial of degree $\geq b+2$ club the monomial as a product of monomial of degree $b+1$ and a monomial of degree less than $2b-(b+1)$ and substitute the monomial of degree $b+1$ with corresponding encryption from K. This will reduce the degree of the polynomial by 1. Repeat this procedure at most b times. For monomials of degree $\leq b-1$ replace by corresponding encryptions from K. Upon completion we get a cipher text of degree at most b hence polynomially bounded in size. Observe that this algorithm is a polynomial time algorithm. If $d=1$ we can optimize this procedure and reduce the size of the set K by keeping encryptions of degree $b+1$ only.

CORRECTNESS: Correctness of $Evaluate()$ stems form correctness of addition and multiplication operation. Suppose, $c_1 = \pi_1 + \sum_{i=1}^{i=N} h_i f_i$ and $c_2 = \pi_2 + \sum_{i=1}^{i=N} g_i f_i$ where c_1, c_2 are of degree at most b. Addition outputs $c \leftarrow c_1 + c_2 = \pi_1 + \pi_1 + \sum_{i=1}^{i=N}(h_i + g_i) f_i$ which decrypts to $\pi_1 + \pi_2$ and is of degree at most b. Upon multiplication we first multiply cipher-text and then perform degree reduction. multiplication gives us something of the form $\pi_1\pi_2 + \sum_{i=1}^{i=N} k_i f_i$ and is of degree at most $2b$ and decrypts correctly. When we perform degree reduction, each monomial of the form $t_1^{e_1}...t_N^{e_N}$ and degree greater than b is replaced by $a_1^{e_1}...a_N^{e_N} + \sum_{i=1}^{i=N} h_i f_i$. Decryption works correctly since the evaluation at the point in the variety is the same and the form of the cipher-text is still $\pi + \sum_{i=1}^{i=N} g_i f_i$ and has degree at most b.

6.2 Proof of Security

Security proof for the case when N was exponential will not work here. [4] provides a proof of bounded security for the scheme. The security proof is based on Ideal membership problem. The maximum number of queries that an adversary can make before security breaks down is $m(\lambda) = \Omega(N)$ [4]. N therefore can be chosen as per requirements.

Definition 9. *(m-time IND-BCPA Security). The m-time IND-BCPA security of a (homomorphic) symmetric-key encryption scheme ε is defined by requiring that the advantage of any ppt adversary $\mathcal{A}$ given by :*
$Adv_{m,\varepsilon,\mathcal{A}}^{IND-BCPA}(\lambda) = Pr[IND-BCPA_{m,\varepsilon,}^{\mathcal{A}}(\lambda)) = True] - 1/2$
is negligible as a function of the security parameter λ . The game $IND-BCPA_{m,\varepsilon}$ is the same as the IND-CPA game with one difference. The difference with the usual IND-CPA security is that the adversary can query its encryption and left-or-right oracles at most $m(\lambda)$ times.

Theorem 4. *Let $\mathcal{A}$ be a ppt adversary against the m-time IND-BCPA security of the scheme. Then there exists a ppt adversary $\mathcal{B}$ against the IM problem such that for all $\lambda \in \mathbb{N}$ we have $Adv_{m,\varepsilon,\mathcal{A}}^{IND-BCPA}(\lambda) = 2Adv_{P,d,b,m,\mathcal{B}}^{IM}(\lambda)$. Conversely, let $\mathcal{A}$ be a ppt adversary against the IM problem. Then there exists a ppt adversary $\mathcal{B}$ against the m-time IND-BCPA security of the scheme such that for all $\lambda \in \mathbb{N}$ we have $Adv_{P,d,b,m,\mathcal{A}}^{IM}(\lambda) = Adv_{m,\varepsilon,\mathcal{B}}^{IND-BCPA}(\lambda)$ [4].*

Proof. Assume $\mathcal{A}$ is an adversary to the IND-BCPA security of the scheme, we construct an IM adversary $\mathcal{B}$ from it. $\mathcal{B}$ is given $\lambda, P, \mathcal{O}$. $\mathcal{O}$ is an oracle that responds to queries of $\mathcal{B}$ and answers with elements in $P_{\leq b} \cap I$, where I is the secret ideal. $\mathcal{B}$ uses $\mathcal{A}$ as follows, if $\mathcal{A}$ queries $Encrypt(m)$, $\mathcal{B}$ responds with $Sample(\mathcal{O}) + m$. When $\mathcal{A}$ sends $\mathcal{B}$ with challenge tuple (m_0, m_1), $\mathcal{B}$ responds by choosing $c \xleftarrow{\$} \{0,1\}$ and sends $f + m_c$ to $\mathcal{A}$. $\mathcal{B}$ declares f to be a member in I if $\mathcal{A}$ answers correctly. In the case when f is in Ideal I advantage of $\mathcal{B}$ is the same as advantage of $\mathcal{A}$ otherwise its advantage is 0 because $\mathcal{A}$ can at most make a guess. Hence, $Adv^{IND-BCPA}_{m,\varepsilon,\mathcal{A}}(\lambda) = 2Adv^{IM}_{P,d,b,m,\mathcal{B}}(\lambda)$.
For the converse, we have an IM adversary $\mathcal{A}$ and we construct an IND-BCPA adversary $\mathcal{B}$ from it. Challenger runs $KeyGen()$, $\mathcal{B}$ is given an access to an oracle $\mathcal{O}$ that returns encryptions of messages queried. When $\mathcal{A}$ queries for elements in the ideal $\mathcal{B}$ queries $\mathcal{O}$ with encryptions of 0 and sends them to $\mathcal{B}$. In the challenge phase $\mathcal{B}$ sends to $\mathcal{A}$ as a challenge tuple $(0, r)$ where r is random in $\mathcal{P}$. If $\mathcal{A}$ responds with the challenge cipher text as an element of ideal the(B) outputs it to be an encryption of 0 else it declares is an encryption of a random element. In this case advantage of $\mathcal{A}$ and $\mathcal{B}$ is the same. Hence,
$Adv^{IM}_{P,d,b,m,\mathcal{A}}(\lambda) = Adv^{IND-BCPA}_{m,\varepsilon,\mathcal{B}}(\lambda)$ In both experiments, it is ensured $\mathcal{O}$ replies queries at most m times.

Corollary. *From GB/IR/IM assumption and theorem 4, the scheme is IND-BCPA secure.*

6.3 KDM security

In order to ensure compactness of the scheme we ouptut encryptions products of the secret key of degree $\leq b+1$. Hence we need to talk about "circular" or KDM security. We just assume KDM security and present a heuristic here that suggests its not a bad assumption to make. Security breaks down when an adversary has a knowledge of $\Omega(m(\lambda))$ or $\Omega(N)$ plain-text cipher-text pairs. We claim that the probability of that knowledge from the public set K is negligible. Consider the distribution of $\{a_1^{e_1}..a_N^{e_N} \mid e_i \geq 0 \forall i \in [1,N] \& \sum_{i=1}^{i=N} e_i \in [1, b+1]\}$ where $a_i \sim U(F_q*) \forall i \in [1,N]$. This distribution can be assumed to be uniform in F_q* (units in F_q). So, for an attack it is best bet to assume everything in the set K as an encryption of a particular constant an devise attack accordingly. Now we argue that in set K probability that an encryption corresponds to an encryption of 1 or equivalently, a constant is negligible. Let $\mathcal{X}_i$ for $i \in F_q*$ denote the random variable that exactly counts number of encryptions of i in K. Note that it is a binomial random variable with parameters $1/(q-1)$ and $\#$ K.

Now probability for the event $\mathcal{X}_c \geq m(\lambda)$ for a constant c is bounded by markov inequality as: $Pr[\mathcal{X}_c \geq m(\lambda)] \leq \#K/(q-1)m(\lambda)$, which is negligible as q is $\Omega(2^\lambda)$ and $\#K$ is polynomial.

6.4 Making the scheme public

The scheme described above can't be made public key if we output many encryptions of "0" because this will enable an adversary to generate $m(\lambda)$ encryptions and produce an attack on the scheme. This is because the scheme is not IND-CPA secure, instead it is IND-BCPA secure.

6.5 What we achieve from this scheme

We get a symmetric, bounded BCPA secure, compact, fully homomorphic encryption without the need to bootstrap.

7 Acknowledgement

I would like to thank my advisors Prof. Palash Sarkar (ISI, Kolkata) and Prof. Ragesh Jaiswal for valuable discussions and insights to the problem.

References

1. Craig Gentry, Shai Halevi *Implementing Gentry's Fully-Homomorphic Encryption Scheme* EUROCRYPT 2011: 129-148
2. Martin Van Dijk, Craig Gentry, Shai Halevi, Vinod Vaikunthanathan *Fully homomorphic encryption over the integers* ADVANCES IN CRYPTOLOGY -2010
3. Ron Rothblum *Homomorphic Encryption: From Private-Key to Public -Key* ECCC -2010
4. Martin R. Albrecht et. al *Polly Cracker, Revisited* ASIACRYPT 2011
5. Zvika Brakerski, Vinod Vaikuntanathan *Efficient Fully Homomorphic Encryption from (Standard) LWE* FOCS 2011: 97-106
6. Craig Gentry, Shai Halevi *Implementing Gentry's Fully-Homomorphic Encryption Scheme* EUROCRYPT 2011: 129-148
7. Zvika Brakerski, Craig Gentry, Vinod Vaikuntanathan *(Leveled) fully homomorphic encryption without bootstrapping* ITCS 2012: 309-325
8. Craig Gentry, Shai Halevi *Fully Homomorphic Encryption without Squashing Using Depth-3 Arithmetic Circuits* FOCS 2011: 107-109
9. Boo Barkee, Deh Cac Can, Julia Ecks, Theo Moriarty, and R. F. Ree *Why you cannot even hope to use Gröbner bases in Public Key Cryptography: An open letter to a scientist who failed and a challenge to those who have not yet failed* Journal of Symbolic Computations 1994, 18(6):497501
10. Massimo Caboara, Fabrizio Caruso, and Carlo Traverso *Lattice Polly Cracker cryptosystems* Journal of Symbolic Computation, 46:534549, May 2011
11. Francoise Levy dit Vehel, Maria Grazia Marinari, Ludovic Perret, and Carlo Traverso *survey on Polly Cracker systems* Coding and Cryptography, pages 285 305. Springer Verlag, Berlin, Heidelberg, New York, 2009
12. Alicia Dickenstein, Noaı Fitchas, Marc Giusti, and Carmen Sessa *The membership problem for unmixed polynomial ideals is solvable in single exponential time* Discrete Appl. Math., 33(1-3):7394, 199
13. Wiliam Fulton *Algebraic curves: An Introduction to Algebraic Curves* Addison Wesley Publishing Company
14. Jonathan Katz and Yehuda Lindell *Introduction to Modern Cryptography: Principles and Protocols* Chapman & Hall/CRC Cryptography and Network Security Series
15. Nigel P. Smart and Frederik Vercauteren *Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes* Public Key Cryptography 2010: 420-443
16. S Gao*Counting Zeros over Finite Fields with Gröbner Bases* Cryptology Eprint Archive
17. Victor Shoup*Sequences of games: a tool for taming complexity in security proofs* Cryptology Eprint Archive