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 information theoretic arguments 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. 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 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 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, all 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

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 public key 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 $\boldsymbol{c} = (c_1, ..., c_i)$ (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 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. Gentry suggests in his work [1] that if a scheme can evaluate its decryption circuit then it can be made into a fully homomorphic encryption using a process called bootstrapping.

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, and a set of polynomials $f_0, ..., f_{m1}$. 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 $G = g_0, ..., g_{m1} \subset I$ is said to be a Gröbner basis of I if for any $f \in I$ there exists $g_i \in G$ such that $LM(g_i) \mid LM(f)$.*

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.

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]. We refer to the framework of [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 query at most $m()$ times. Adversary has to output a 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 query 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 query at most $m()$ times. Challanger than challenges $\mathcal{A}$ with a random polynomial in $P_{\leq b}$, 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 also suggests that it is reasonable to assume:

Definition 7. *(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.

5 Our first Construction

Fix the ring as $R = F_q[t_1, t_2, ..., t_N]/(t_1^q - t_1, .., t_N^q - t_N)$. 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 $\mathcal{G} = \{g_k \mid k \in \mathcal{K}\}$. Key space, $\mathcal{K}$ has atleast $\omega(2^\lambda)$ elements. Heuristically, one possible construction is assuming they(Alice and Bob) have a prior key exchange to a deterministic secure(at least λ bit secure) symmetric key Encryption scheme($Enc_K(*)$)(AES let's say) then, $g(n) = H(Enc_K(n))$ for $n \in [1, N]$ where $H(*)$ is a hash(compression) function and maps encryption to F_q. Let's us describe our first candidate scheme $\pi = (KeyGen, Enc, Dec, Eval)$ now.

$KeyGen(\lambda)$: Output a secret function g that allows to compute $g(n)$ for $n \in [1, N]$ in polynomial time. 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

F_2

Theorem 1. *Let R 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(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 $\boldsymbol{R} = (R_1, R_2, ..., R_m)$. From this set, select another set of $k \in [1, m]$ numbers and order them in ascending order and call this vector $\boldsymbol{L} = (L_1, ..., L_k)$. To encrypt a bit b, output:
$b + \sum_{i=0}^{i=k}(t_{L_i} - g(L_i)) \times f_i(t_{R_1}, ..., t_{R_m})$
where, $f_i(t_{R_1}, ..., t_{R_m})$'s are random polynomials in the ring which depends on at most m indeterminates $(t_{R_1}, ..., t_{R_m})$. 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, $(t_1 t_5 t_{11} + t_2) \times (t_7 + t_5) = t_1 t_5 t_{11} t_7 + t_2 t_7 + t_1 t_5 t_{11} + t_2 t_5$. This makes cipher-text is linear in all indeterminates.

Algorithm 1 Encrypt(g,b)

Input: Secret key function g and a bit b

Output: A cipher text $c(t_1, .., t_N)$, polynomial in at most m indeterminates which encrypts b

Select m numbers from $[1, N]$. m is $O(log(\lambda))$. Sort them in ascending order, call this random number vector $\boldsymbol{R} = (R_1, R_2, ..., R_m)$

Select $k \in [1, m]$ randomly.

Select k random numbers from $\boldsymbol{R}$ and store them as $\boldsymbol{L} = (L_1, ..., L_k)$ in ascending order.

$c(t_1, .., t_N) \leftarrow b + \sum_{i=0}^{i=k}(t_{L_i} - g(L_i)) \times f_i(t_{R_1}, ..., t_{R_m})$ where $f_i(t_{R_1}, ..., t_{R_m})$ are randomly chosen over the ring and depends on at most selected m indeterminates.

return $c(t_1, .., t_N)$

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

Algorithm 2 *Decrypt*$(g, c(t_1, ..., t_N))$

Input: Secret key function g and a cipher-text polynomial $c(t_1, .., t_N)$

Output: Decryption of $c(t_1, ..., t_N)$

$b \leftarrow c(g(1), g(2), ..., g(N))$

return b

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 $\times$ 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 $poly(\lambda)$ in that case the depth that we can solve is $loglog(poly(\lambda)) - log(m)$. But this is an overkill. If we somehow encrypt such that fresh-cipher text has at most $O(1)$ length, we can solve $O(log(\lambda))$ deep circuits. This is because at each level the length of the cipher text squares so at level d it will be c^d and for this to be polynomially bounded we require d to be $O(log(\lambda))$, for a constant c.

One important feature is, given a cipher-text tuple of $O(1)$ fresh cipher-texts, we can evaluate circuits of any depth on this vector, 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

Looking at a cipher-text $c(t_1, .., t_N)$, a polynomial in at most m variables, If an adversary knows nothing about any of the $g(i)$'s for an $i \in [1, N]$ for which t_i appear in the cipher text, he can't predict the bit with a probability better than a toss of a coin.

Now let's consider a CPA security game, We have an adversary A who has been given an access to encryption oracle. He then outputs two messages (m_0, m_1) and challenger outputs an encryption of randomly chosen message. Adversary still continues to have an access to the encryption oracle. Adversary has to guess which message was encrypted m_0 or m_1. If guesses correctly the experiment outputs 1 else it outputs 0. Let's denote the event of experiment outcome as $PrivK^{cpa}_{A,\pi}(\lambda)$.

Security lies in the idea that suppose he uses encryption oracle $q(\lambda)$ times for a polynomial $q(*)$(don't confuse with the characteristic of the field). Each cipher text would have dependency on at most m variables. Suppose these queries

enable adversary to learn $m \times q(\lambda)$ evaluations of g for integers i for $i \in [1, N]$ and t_i appearing in queries returned by the oracle. The claim is that only with a negligible probability will the adversary know atleast one g value $g(i)$ for t_i appearing in the challenge cipher text. Since challenge cipher text is dependent on new set of variables and adversary has no knowledge of the g function values for those integers i for which t_i appear in the challenge cipher text, he can't do much more than a random guess.

Suppose the adversary learns $m \times q(\lambda)$ values of the g function. Let the set I denote the set of integers i such that $g(i)$ is known to the adversary. The cardinality of this set is at most $m \times q(\lambda)$. Probability that the challenge cipher text depends on atleast one variable t_i for $i \in I$ is $1 - \frac{(N-mq(\lambda))^m}{N^m}$, which is less than $\frac{m^2 q(\lambda)}{N}$. This is because if we calculate probability using the argument that any one of the m variables in cipher text can come from the set of $mq(\lambda)$ variables, while others can come from anywhere, this will give us the probability $\frac{m^2 q(\lambda)}{N}$, but counting in this manner involves doubly counted cases. Suppose adversary tells correctly which message was encrypted if he knows the value of any(atleast one) $g(i)$ for which t_i appear in the challenge cipher text with probability 1. Otherwise, he just makes a guess and answers correctly with a probability $\frac{1}{2}$.

Lets calculate the advantage of such an adversary:

$Pr[PrivK^{cpa}_{A,\pi}(\lambda) = 1] = \frac{1}{2}(\frac{(N-mq(\lambda))^m}{N^m}) + (1 - \frac{(N-mq(\lambda))^m}{N^m}).1$

$Pr[PrivK^{cpa}_{A,\pi}(\lambda) = 1] = 1 - \frac{1}{2}(1 - \frac{mq(\lambda)}{N})^m$

Using the approximation $(1-x)^n \approx 1 - nx$ for small x we get:

$Pr[PrivK^{cpa}_{A,\pi}(\lambda) = 1] \approx \frac{1}{2} + \frac{m^2 q(\lambda)}{2N}$

Using the above equation or the argument on double counting we get,

$Pr[PrivK^{cpa}_{A,\pi}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda)$

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 information theoretic arguments rather than basing them on problems like Gröbner basis/ Ideal Membership/ Ideal Remainder which yield us at most bounded security.

Implementations 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 depth.

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.

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 and public Key. 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. Here it is assumed that $q = q(\lambda)$ is $\Omega(2^\lambda)$. The monomials in this ring take 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 a symmetric key scheme and later on describe how it can be made into a public key scheme. Suppose Alice and Bob wants to communicate secretly. As before, they agree on a secret random function $g : [1, N] \to F_q$. Suppose, this function is λ bit secure. The ideal we work on is $I = (t_1 - g(1), .., t_N - g(N))$. Check that one Gröbner basis G for ideal I is $\{t_i - g(i) \mid i \in [1, N]\}$. Since N is polynomial g can be replaced by a random string. Encryption of bit $bit \in F_q$ is an element of the ideal coset $bit + I$. Plain text space is F_q. Let there be a constant $b > 1$(consistent with the notation used in preliminaries). Cipher-texts are polynomials in the ring with a degree at most b. Note that for our scheme d(degree of polynomial in the Gröbner basis) is 1. Given a polynomial $f \in P$, f mod G= $f(g(1), .., g(N))$.

Let us describe the algorithms:

$KeyGen(\lambda)$ A secret random function $g : [1, N] \to F_q$ is chosen as the secret key. This function is assumed to be λ secure. We also publish a public set which will be used by evaluate algorithm to ensure compactness of cipher-texts. This set is denoted by K. It is the set which has the encryptions(refer *Encrypt*) of all $O(\binom{N+b}{b})$ elements of the set $\{g(1)^{e_1}..g(N)^{e_N}) \mid e_i \in [0, b+1] \forall i \in [1, N]$ & $\Sigma_{i=1}^{N} e_i = b + 1\}$

Encrypt(g, π): Take a random polynomial in the ring, whose degree is less than or equal to b. Length of this polynomial is $O(\binom{N+b-1}{b-1})$(count possible number of monomials). Since N is a polynomial and b is $O(1)$, length is polynomially bounded. Denote this polynomial by $f(t_1, .., t_N)$. If $f(g_1, .., g_N) = x$ output $c(t_1, .., t_N) = f(t_1, .., t_N) - x + \pi$.
Select randomly $f \in P_{\leq b}$
Output c=$f - f \mod G + \pi$

Decrypt$(g, c(t_1, .., t_N))$ Output $c(g(1), .., g(N))$

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 $g(1)^{e_1}..g(N)^{e_N}$. 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. 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.

CORRECTNESS: Correctness of add and mul operations follow from the ring operations. I=$(t_1 - g(1), .., t_N - g(N))$. Encryption of $\pi \in \pi + I$. Correctness can be seen from the fact that $\pi_1 + i_1 + \pi_2 + i_2 \in \pi_1 + \pi_2 + I$ for all $i_1, i_2 \in I$ and $(\pi_1 + i_1).(\pi_2 + i_2) \in \pi_1.\pi_2 + I$ for all $i_1, i_2 \in I$. It only remains to prove that degree reduction works correctly. Assume that there is a cipher text $c(t_1, .., t_N)$ which encrypts π, so that $c(g(1), .., g(N)) = \pi$. As long as we replace any monomial in $M(c)$ with a polynomial that evaluates to same thing when evaluated on $(g(1), .., g(N))$ we still get a valid cipher text. For every $m \in M(c)$. When $m = t_1^{e_1}..t_N^{e_N}$, on replacing m by $g(1)^{e_1}..g(N)^{e_N} + Encrypt(0)(t_1, .., t_N) = Encrypt(g(1)^{e_1}..g(N)^{e_N})(t_1^{e_1}...t_N^{e_N})$ we still get a valid cipher text but with a reduced degree.

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 $m(\lambda) = \Omega(N)$. N therefore can be chosen as per requirements.

Definition 8. *(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 :*
$Adv^{IND-BCPA}_{m,\varepsilon,\mathcal{A}}(\lambda) = 2Pr[IND - BCPA^{\mathcal{A}}_{m,\varepsilon,}(\lambda)) = True] - 1$
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 2. *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* $Adv^{IND-BCPA}_{m,\varepsilon,\mathcal{A}}(\lambda) = 2Adv^{IM}_{P,d,b,m,\mathcal{B}}(\lambda)$. *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* $Adv^{IM}_{P,d,b,m,\mathcal{A}}(\lambda) = Adv^{IND-BCPA}_{m,\varepsilon,\mathcal{B}}(\lambda)$ *[4].*

6.3 KDM security

In order to ensure compactness of the scheme we ouptut encryptions of $b+1$ degree products of the secret key. 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 $\Omega(N)$ plaintext cipher text pairs. We claim that the probability of that knowledge from the public set K is negligible. Consider the distribution of $\{g(1)^{e_1}..g(N)^{e_N} \mid e_i \geq 0 \forall i \in [1,N] \& \Sigma_{i=1}^{i=N} e_i = b+1\}$ where $g(i) \sim U(F_q) \forall i \in [1,N]$ has a spike at 0 with a uniform distribution on F_q*(units in F_q). This is because 0 multiplied with anything gives 0. So, for an attack it is best bet to assume everything in the set K as an encryption of 0. Now we argue that in set K probability that an encryption corresponds to an encryption of 0 is negligible. Let $\mathcal{X}_i(j)$ for $i \in F_q \& j \in$ [1,# K] denote the event that exactly j encryptions in the set K are encryptions of i.

Probability that atleast one of the elements of K correspond to an encryption of $0 \in F_q$ when q is $\Omega(2^\lambda)$ is:
$Pr[\mathcal{X}_0(\geq 1)]$ $1-(1-1/q)^N$.
$Pr[\mathcal{X}_0(\geq 1)] \approx N/q$.
Which is negligible. This is calculated by the fact that with negligible probability any one of $g()$ value is 0.

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