Fully Homomorphic Encryption

Boaz Barak

February 9, 2011

Achieving fully homomorphic encryption, under any kind of reasonable computational assumptions (and under any reasonable definition of "reasonable"..), was a holy grail of cryptography for many years until finally achieved by Craig Gentry in 2009. In these lectures we'll describe a somewhat simplified variant of Gentry's construction obtained by van Dijk, Gentry, Halevi and Vaikuntanathan (with another slight simplification suggested by Sushant Sachdeva, and was also independently observed by Ivan Damgard).

1 Definitions

We recall the definition of fully homomorphic encryption:

Definition 1. We say that a quadruple of p.p.t algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is a *strongly fully homomorphic encryption scheme* (FHE for short) if $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is semantically secure and in addition for every $t = \mathrm{poly}(n)$ and polynomial size circuit C taking t bits as input, every $b_1, \ldots, b_t \in \{0,1\}$ and $c_1, \ldots, c_t$ output by $\mathsf{Enc}_{pk}(b_1), \ldots, \mathsf{Enc}_{pk}(b_t)$, the distributions $\mathsf{Eval}_{pk}(C, c_1, \ldots, c_t)$ and $\mathsf{Enc}_{pk}(C(b_1, \ldots, b_t)$ are statistically indistinguishable.

It's a relatively straightforward exercise to show that it is enough to have an evaluation algorithm for AND and XOR, or equivalently multiplication and addition modulo 2. That is, the following is an equivalent alternative definition of FHE:

Definition 2. We say that a quadruple of p.p.t algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$ is a *strongly fully homomorphic encryption scheme* (FHE for short) if $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is semantically secure and in addition for every $b, b' \in \{0,1\}$ and c, c' output by $\mathsf{Enc}_{pk}(b)$ and $\mathsf{Enc}_{pk}(b')$ respectively:

- The distributions $\mathsf{Mult}_{pk}(c, c')$ and $\mathsf{Enc}_{pk}(b \wedge b')$ are statistically indistinguishable.
- The distributions $\mathsf{Add}_{pk}(c, c')$ and $\mathsf{Enc}_{pk}(b \oplus b')$ are statistically indistinguishable.

For this class, we define two distributions X, Y over $\{0,1\}^n$ are statistically indistinguishable if for every function $f : \{0,1\}^n \to \{0,1\}$, $\|E[f(X)] - E[f(Y)]\| < 2^{-n^{1/1000}}$.

Recall that an encryption scheme is semantically secure if the two distributions $(pk, \mathsf{Enc}_{pk}(0))$ and $(pk, \mathsf{Enc}_{pk}(1))$ are computationally indistinguishable (e.g., any circuit of polynomial size will succeed in distinguishing them with bias less than one over any polynomial; if you want, you can replace "any polynomial" with some fixed super-polynomial function such $2^{-n^{1/1000}}$).

The definitions for private key encryption are analogous, but now $\mathsf{Eval}, \mathsf{Add}, \mathsf{Mult}$ don't get the key as input. As mentioned in class (and is in fact an easier variant of the homework exercise), one can easily transform a private key FHE into a public key FHE, letting the public key be encryptions of zero and one.

From now on, we'll focus on constructing a scheme satisfying Definition 2.

2 Construction of a "noisy homomorphic" encryption scheme.

Our first step will be to construct a weaker notion than FHE that we'll call a "noisy homomorphic encryption scheme". This will also be an encryption scheme with the Add and Mult algorithms, but will satisfy a relaxed notion of homomorphism. It will be easier to first construct the scheme, and then define precisely the notion of homomorphism it satisfies. We remark that our notion of noisy homomorphism will imply weak homomorphism (i.e., compact encryption) with respect to some subclass of circuits (specifically arithmetic circuits modulo 2 with depth up to $n/100$ where n is our security parameter).

2.1 LDN Assumption

The encryption scheme will be based on the following computational problem. Let $N = PQ$ be equal to the product of two large primes, and suppose that you are given various random products $R_1P \pmod N, \ldots, R_tP \pmod N$ where the R_i's are chosen randomly in $[Q]$.

In this case, recovering P is easily done by computing the gcd of, say, R_1P and R_2P since with good probability P will be the only common factor between them. But the gcs algorithm is extremely sensitive to noise, and so it is not clear how to adapt this to the case when you are given $R_1P + E_1 \pmod N, \ldots, R_tP + E_t \pmod N$, where the E_i's are noise factors that are chosen in some interval $[-\mathbf{E}, + - \mathbf{E}]$ for a parameter $\mathbf{E} \ll N$. This motivates us to making the following assumption:

Learning divisor with noise (LDN) assumption. Let P a random n bit prime, Q a random n^4 bit prime, and let $N = PQ$ and $\mathbf{E} \geq 2^{n^{0.1}}$. Then, for every $t = \text{poly}(n)$ there is no polynomial time algorithm that on input N and $X_1, \ldots, X_t \in \mathbb{Z}_N$ can distinguish between case **(I)** the X_i's are chosen independently at random from $\mathbb{Z}_N$, and **(II)** $X_i = PR_i + 2E_i \pmod N$ where R_i is chosen independently at random from $\mathbb{Z}_Q$ and E_i is chosen independently at random from $[-\mathbf{E}, +\mathbf{E}]$.

Some notes are in order:

1. We use small letters such as n, t for values that are polynomial in the security parameter (which we often denote by n), and capital letters such as N, P, Q, R for large numbers that have $\text{poly}(n)$ digits (and hence are exponential in n).

2. We denote by $\mathbb{Z}_N$ the set $\{0, \ldots, N-1\}$ and by $\mathbb{Z}*_N$ the set $\{X \in \mathbb{Z}_N : gcd(X, N) = 1\}$. If N is prime then $\mathbb{Z}_N^* = \{1, \ldots, N-1\}$. When speaking about a random X, it usually won't matter if we take it from $\mathbb{Z}_N$ or $\mathbb{Z}_N^*$ since $|\mathbb{Z} *_N |/|\mathbb{Z}_N| \ll |\mathbb{Z}_N|$.

3. The assumption can be seen to be monotone in $\mathbf{E}$. That is, increasing $\mathbf{E}$ only makes the problem of distinguishing between the two cases harder. (The intuition is that the distinguisher can always add more noise to the inputs on its own.)

4. We made the noise even (i.e. $2E_i$ instead of $2E_i$) for convenience, but the two choices are equivalent - exercise!. See footnote for hint[1]

5. Obviously the LDN assumption is stronger than the assumption that factoring N is hard. There is a variant of the LDN assumption that is still useful for FHE but is not known to imply that factoring is hard.

[1] **Hint:** Given a number of the form $RP + E$ if you multiply it by two you get $2RP + 2E$ but since Q is odd, if R is uniform over $\mathbb{Z}_Q$ then $2R$ is uniform over $\mathbb{Z}_Q$ as well.

2.2 The noisy homomorphic encryption scheme

We now describe the noisy homomorphic encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$. It will be a private key scheme (since it will be easy to convert it eventually to public key). We will assume however that it has *public parameters*— which are values that are generated by the key generation algorithm Gen and made public and can be used by all algorithms. Formally, such public parameters are never needed since we can always append them to the encryption, but it makes for cleaner notation to assume them.

Key We choose P to be a random n bit prime, and Q to be a random n^4 bit prime, $N = PR$. We keep P secret, and can publish N as a public parameter.

Encryption For $b \in \{0,1\}$ we let $\mathsf{Enc}_P(b) = \mathsf{Enc}_P^{2^{\sqrt{n}}}(b)$, where $\mathsf{Enc}_P^{\mathbf{E}}(b)$ is defined as follows: choose $R \leftarrow_{\mathrm{R}} \mathbb{Z}_Q$ and $E \leftarrow_{\mathrm{R}} [-\mathbf{E}, +\mathbf{E}]$, and output $X = RP + 2E + b \pmod N$.

Decryption To decrypt X, output $X - \lfloor X/P \rceil P \pmod 2$. ($\lfloor x \rceil$ denotes the integer closest to x, breaking ties, say, downwards.)

Security and correctness of scheme. The choice $2^{\sqrt{n}}$ for the parameter $\mathbf{E}$ makes the scheme both correct and secure. More generally, as long as $\mathbf{E} \ll P$ (say $\mathbf{E} < 2^{0.9n}$) then decryption will succeed, since $X - \lfloor X/P \rceil P$ will equal $2E+b$. As long as $\mathbf{E} > 2^{n^{0.1}}$ then under the LDN assumption the scheme will be secure.

Noisy/weak homomorphism. We have the operations Add and Mult defined simply as $\mathsf{Add}(X, X') = X + X' \pmod N$ and $\mathsf{Mult}(X, X') = X \cdot X' \pmod N$.

Let $\mathcal{E}^{\mathbf{E}}(b)$ denote the set of possible encryptions of b with parameter $\mathbf{E}$. That is $\mathcal{E}^{\mathbf{E}}(b) = \{X : X = RP + 2E + b \pmod N, R \in \mathbb{Z}_Q, E \in [-\mathbf{E}, +\mathbf{E}]\}$. Note that if $\mathbf{E}' \geq \mathbf{E}$ then $\mathcal{E}^{\mathbf{E}}(b) \subseteq \mathcal{E}^{\mathbf{E}'}(b)$. Intuitively, we think of the set $\mathcal{E}^{\mathbf{E}}(b)$ as the set of ciphertexts with noise parameter $\mathbf{E}$.

If the scheme was fully homomorphic then we'd have the following condition for $\mathbf{E} = 2^{\sqrt{n}}$: if $X \in \mathcal{E}^{\mathbf{E}}(b)$ and $X' \in \mathcal{E}^{\mathbf{E}}(b')$, then $\mathsf{Add}(X, X')$ is in $\mathcal{E}^{\mathbf{E}}(b \oplus b')$ (and moreover it's uniformly distributed in that set), and similarly $\mathsf{Mult}(X, X')$ is uniform in $\mathcal{E}^{\mathbf{E}}(b \wedge b')$. This is not necessarily true, but we can show that the results of Add and Mult are in the corresponding set but with a somewhat larger noise parameters. That is, we can prove that for every $\mathbf{E}, \mathbf{E}'$

If $X \in \mathcal{E}^{\mathbf{E}}(b)$ and $X' \in \mathcal{E}^{\mathbf{E}'}(b')$ then
(i) $\mathsf{Add}(X, X') \in \mathcal{E}^{2(\mathbf{E}+\mathbf{E}')}(b \oplus b')$ and
(ii) $\mathsf{Mult}(X, X') \in \mathcal{E}^{5\mathbf{E}\mathbf{E}'}(bb')$.

As a result, if we start with m ciphertexts with noise parameter $2^{\sqrt{n}}$ then we can add and multiply them and as long as we don't take a product of more than say $n^{1/10}$ of them then we'll still get ciphertexts of noise $\ll 2^n$ (and hence we can decrypt them). This implies the following consequence, that would be useful below:

Define an *arithmetic circuit* to be a circuit C taking m inputs and having one output that consists of only multiplication gates, addition gates, and the constants 1 and 0. We define $\mathcal{P}(C)$ to be the polynomial mapping $\mathbb{Z}^m$ to $\mathbb{Z}$ that C computes. Note that if we think of C as a Boolean circuit with multiplication gates as $\wedge$ and addition gates as $\oplus$ then

$$C(b_1, \ldots, b_m) = \mathcal{P}(C)(b_1, \ldots, b_m) \pmod 2$$

For a polynomial $f : \mathbb{Z}^m \to \mathbb{Z}$ and $M \geq 0$ we define $|f|_M$ to be the maximum over all $x_1, \ldots, x_m$ satisfying $|x_i| \leq M$ of $|f(x_1, \ldots, x_m)|$.

Lemma 3. *Let $P, \mathbf{E}$ be the parameters of our encryption scheme, and let C be an arithmetic circuit such that $|\mathcal{P}(C)|_{2\mathbf{E}+1} < P/10$. Then, for every $b_1, \ldots, b_m \in \{0,1\}$, if we let $X_i = \mathsf{Enc}_P(b_i)$ and evaluate the circuit C on $X_1, \ldots, X_m$, using* Add *and* Mult *for the gates to obtain a result X^*, then we'll have that*

$$\mathsf{Dec}_P(X^*) = C(b_1, \ldots, b_m) \, (= \mathcal{P}(C)(b_1, \ldots, b_m) \pmod 2)$$

Proof. We know that for every $i = 1..m$, $X_i = R_i P + 2E_i + b_i$ where $|E_i| \leq \mathbf{E}$. Let $\mathcal{P} = \mathcal{P}(C)$, then because taking products and sums of multiples of P still results in a multiple of P, we know that

$$\mathcal{P}(X_1, \ldots, X_m) = KP + \mathcal{P}(2E_1 + b_1, \ldots, 2E_m + b_m)$$

for some integer K. Moreover, since N is a multiple of P this is still true if we reduce modulo N, which means that

$$X^* = \mathcal{P}(X_1, \ldots, X_m) \pmod N = K'P + \mathcal{P}(2E_1 + b_1, \ldots, 2E_m + b_m)$$

for some integer K'. Under our condition we know that $|\mathcal{P}(2E_1 + b_1, \ldots, 2E_m + b_m)| < P/10$ and hence

$$X^* - \lfloor X^*/P \rceil \, P = X^* - K'P = \mathcal{P}(2E_1 + b_1, \ldots, 2E_m + b_m)$$

Now by the same reasoning as above

$$\mathcal{P}(2E_1 + b_1, \ldots, 2E_m + b_m) = 2K'' + \mathcal{P}(b_1, \ldots, b_m)$$

for some integer K'' and hence

$$X^* - \lfloor X^*/P \rceil \, P \pmod 2 = \mathcal{P}(b_1, \ldots, b_m) \pmod 2$$

□

This lemma may seem hard to use, since it requires bounding $|\mathcal{P}(C)|_{2\mathbf{E}+1}$, but we'll only use the following very simple observation:

If $f(x_1, \ldots, x_m)$ is an integer polynomial of degree d where all its monomials have coefficients of magnitude at most C, then $|f|_M \leq C \cdot m^d \cdot M^d$.

3 Making it fully homomorphic

Clean and ReRand To make the scheme above fully homomorphic we'll add two operations to it:

- $\mathsf{Clean}(X)$ will take as input a ciphertext in $\mathcal{E}^{2^{n^{0.9}}}(b)$ and output a ciphertext in $\mathcal{E}^{2^{n^{0.3}}}(b)$. That is, it reduces the noise of the ciphertext.
- $\mathsf{ReRand}(X)$ will take as input a ciphertext in $\mathcal{E}^{2^{n^{0.4}}}(b)$ and output a ciphertext that distributed statistically close to the uniform distribution over $\mathcal{E}^{2^{\sqrt{n}}}(b)$, that is, $\mathsf{ReRand}(X) \approx \mathsf{Enc}(b)$.

Fully homomorphic encryption Together Clean and ReRand imply a fully homomorphic encryption scheme: we just change the definition of Mult and Add to apply Clean and ReRand as follows:

- $\mathsf{Mult}(X, X') = \mathsf{ReRand}(\mathsf{Clean}(X \cdot X' \pmod N))$.

- $\mathsf{Add}(X, X') = \mathsf{ReRand}(\mathsf{Clean}(X + X' \pmod N))$.

Now, given two ciphertexts X, X' in the range of the encryption algorithm encrypting b and b' respectively (i.e., $X \in \mathcal{E}^{2^{\sqrt{n}}}(b)$ and $X' \in \mathcal{E}^{2^{\sqrt{n}}}(b')$), the value $Y = X \cdot X' \pmod N$ will be in $\mathcal{E}^{5 \cdot 2^{2\sqrt{n}}}(b \wedge b') \subseteq \mathcal{E}^{2^{n^{0.9}}}(b \wedge b')$, and hence $Z = \mathsf{Clean}(Y)$ will be in $\mathcal{E}^{2^{n^{0.3}}}(b \wedge b') \subseteq \mathcal{E}^{2^{n^{0.4}}}(b \wedge b')$, meaning that $\mathsf{ReRand}(Z)$ will be statistically indistinguishable from an encryption of $b \wedge b'$.

Same analysis applies to Add.

Note: As you can see, we have considerable "slackness" in the parameters of ReRand and Clean, I chose these values to demonstrate that the parameter choice here needs to be done somewhat carefully, but it's not extremely fragile.

Our goal is now to get both Clean and ReRand. Clean is really the important one among those— the rerandomization property can often be achieved for many encryption schemes.

Getting ReRand We'll briefly mention how one can get ReRand, leaving verifying the details to the homework exercise. Our input is a ciphertext of the form $X = RP + 2E + b$ where $|E| \leq 2^{n^{0.4}}$. We want to transform it into $X' = R'P + 2E' + b$ where R' is uniform in $[Q] = [N/P]$ and E' is uniform in $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$.

- *Rerandomizing noise:* if we just wanted to rerandomize the noise we could just choose E'' uniformly in $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$, and add $2E''$ to X. If we look at $E' = E + E''$ then this is distributed uniformly in the interval $[-2^{\sqrt{n}}, +2^{\sqrt{n}}] + E$ which is within $2^{n^{0.4}}/2^{\sqrt{n}} = negl(n)$ statistical distance to the uniform distribution over $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$.
- *Rerandomizing multiple:* rerandomizing R is a bit more tricky. The idea is the following: suppose we have at our disposal many, say $X_1, \ldots, X_m$ for $m = n^6$, random encryptions of 0 with small noise (less than $2^{n^{0.4}}$). Then we will choose at random a subset $S \subseteq [m]$ and will look at the ciphertext $X'' = X + \sum_{i \in S} X_i$. This is still an encryption of 0 with at most $m2^{n^{0.4}}$ noise, and the corresponding multiple is just

$$R + \sum_{i \in S} R_i \pmod Q$$

where $X_i = PR_i + 2E_i$. We then use the following lemma (variant of what's known as "leftover hash lemma"):

Lemma 4. *Let Q be a k bit prime and suppose that $R_1, \ldots, R_m$ are chosen at random in $\mathbb{Z}_Q$ where $m > 10k$. Then with probability at least $1 - 2^{-k/10}$ over the choice of $R_1, ..., R_m$, if we fix them and consider the random variable $R = \sum_{i \in S} R_i \pmod Q$, where S is a random subset of $[m]$, then R is within $2^{-k/10}$ statistical distance to the uniform distribution over $\mathbb{Z}_Q$.*

You can prove it like the homework exercise as follows. Fix some value $\alpha \in \mathbb{Z}_Q$: we want to argue that with very high probability (enough to take union bound over all of $\mathbb{Z}_Q$), if we choose the random $R_1, \ldots, R_m$ randomly and fix them, the random variable R will satisfy $|\Pr[R = \alpha] - 1/Q| < 2^{-2k}$ (or some similar bound) where this probability is over the choice of the set $S \subseteq [m]$ that determines R. We can do so by defining for every nonempty subset S of $[m]$ a random variable X_S over the choice of $R_1, \ldots, R_m$ that is 1 if $\sum_{i \in S} R_i \pmod Q = \alpha$, and then showing $E[X_S] = 1/Q, E[X_S X_T] = 1/Q^2$ and using the Chebychev inequality (using the fact that $2^m \gg 2^k$).

- *Putting it all together:* We combine these to get ReRand as follows: as part of the public parameters (or concatenated to any encryption) we add ciphertexts $X_1, \ldots, X_m$ where $X_i = R_i P + 2E_i$ with $R_i \leftarrow_R [Q]$ and $E_i \leftarrow_R [-2^{n^{0.4}}, +2^{n^{0.4}}]$. Then to rerandomize X we choose a random subset S of $[m]$, and E'' at random from $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$ and output

$$X' = X + \sum_{i \in S} X_i + 2E'$$

Getting Clean using "wishful thinking" We now tackle the bigger problem - how to get the cleanup procedure Clean. This is very challenging, since up until this point it seems that any operation we do on ciphertexts, adding/multiplying/rerandomizing etc..., only increases the noise. In fact, it seems somewhat counterintuitive that you could decrease the noise without knowing the secret key, since if you could decrease it too much then you would be able to find out the plaintext!

Nevertheless, we'll show it may be possible to clean up the ciphertext, at least if we happened to be very lucky and the encryption scheme satisfies a certain property.

Let us consider the decryption algorithm Dec. The algorithm takes as input the secret key P and a ciphertext X and outputs the corresponding bit b. Since P and X are in the end represented by bits, Dec is just a function mapping $\{0,1\}^m$ to $\{0,1\}$ (where $m = n + n^5$ is the length of this description; this is not the same number m we used in the ReRand operation).

This is an *efficient* function, and so it can be computed by a polynomial size Boolean circuit C, and we can assume that the gates of this circuit are only $\cdot$ and $\oplus$ (plus the constants $0, 1$) since they are universal. Now suppose that we are lucky $|\mathcal{P}(C)|_{2^{n^{0.1}}} < 2^{n^{0.3}}$. For example, this can happen if $\mathcal{P}$ happens to be a polynomial with 0/1 coefficients and degree at most $n^{0.1}$. We claim that in this case we can run the Clean operation as follows:

Recall that we're given an input $X = RP + 2E + b$ where $|E| \leq 2^{n^{0.9}}$ and our goal is to come up with $X' = R'P + 2E' + b$ such that $|E'| \leq 2^{n^{0.3}}$. We are going to do the following:

1. We change the scheme to include $Y_1, \ldots, Y_n$ in the public parameters where $Y_i = \mathsf{Enc}_P^{2^{n^{0.1}}}(P_i)$ with P_i being the i^{th} bit of P. That is, we include an encryption of P in the public parameters, using noise value $2^{n^{0.1}}$ which is smaller than the standard parameter, but still big enough to ensure security.
2. The Clean operation will be defined as follows. Recall that we are given an encryption X which is a number modulo the n^5 bit number N. X encrypts some bit b with noise parameter $2^{n^{0.9}}$ and we want to come up with an encryption X' encrypting the same b with noise parameter $2^{n^{0.3}}$.

   We take $Y_1, \ldots, Y_n$ from the public parameters, and define $Y_{n+1}, \ldots, Y_m$, (where $m = n + n^5$) according to the bits of the ciphertext X. That is, Y_{n+1} is 1 if the first bit of X is 1 and 0 otherwise, etc.. Note that we can think of the number 1 also as an encryption of 1 (after all $1 = 0 \cdot P + 2 \cdot 0 + 1$) and similarly we can think of the number 0 also as an encryption of 0.

   Therefore, we now have ciphertexts $Y_1, \ldots, Y_m$ that are encryptions of the string $P \circ X$ where $\circ$ denotes concatenation. Moreover, these ciphertexts Y_i have very low noise! That is, each one of them has noise at most $2^{n^{0.1}}$. (Where our goal is at the end to get a ciphertext of noise $2^{n^{0.3}}$.)

Now we know that $\mathsf{Dec}(P \circ X) = b$, and so if we run the circuit C on the encryptions $Y_1, \ldots, Y_m$ we should get a ciphertext X' encrypting b which is exactly what we wanted! (This argument is so beautiful it deserves all the exclamation marks it gets...

The only thing left to verify is that the noise of X' is not that large. The idea is that because the circuit is simple, and we started from ciphertexts with small noise then we will get a ciphertext with not too large noise. Specifically, the noise level we'll get, by the same argument as above will be at most $|\mathcal{P}(C)|_{2 \cdot 2^{n^{0.1}}+1}$.

A relatively minor issue Unfortunately, it turns out that CPA security does not guarantee that it is secure to encrypt the secret key with itself (exercise...). We overcome this issue by using a common cryptographic technique— making an assumption: we'll assume that even given oracle access to the encryption oracle, one cannot distinguish an encryption of the secret key and an encryption of the all zero string.

This notion is called *circular security* and it is a subclass of a more general notion of *key dependent message (KDM) security*. While there are examples of CPA (and even CCA) secure schemes that are not circular secure, there are no known attacks against natural cryptosystems (e.g., El-Gamal etc..) and so it seems a reasonable assumption to assume that they are circular secure. In recent years, a few encryption schemes were proven to be circular secure (and satisfy some notions of KDM security as well) under relatively standard assumptions. It is also easy to construct KDM secure schemes in the random oracle model, and there are ways to try to combine this construction with the homomorphic scheme to make it even more likely it is circular secure.

In any case, we will assume this scheme is circular secure. Hopefully at some point someone will manage to prove that it is, and get rid of this assumption.[2]

As I already mentioned, the question of getting any plausible homomorphic encryption scheme, even with only heuristic security such as the random oracle model, was open for 30 years, so we shouldn't complain too much even if the solution uses somewhat non-standard assumptions.

A major issue The major issue is that we had no reason to believe that our circuit have small norm. Generally a circuit over $\{0,1\}^m$ is expected to have polynomial degree about $m \sim n^5$, and indeed I believe one can verify that the decryption circuit actually computes a polynomial of degree at least $n/100$, and will actually satisfy $|\mathcal{P}(C)|_1 > 2^{n/100}$. So we're off by a polynomial factor in the exponent.

We will tackle this issue by making an additional tweak to the encryption scheme, intended to "squash" the decryption circuit and make it of smaller degree.

[2] Even without this assumption one can get a limited homomorphic encryption scheme, where the public key grows with the depth of the circuit, see the papers and Gentry's thesis.