6.889: New Developments in Cryptography February 8, 2011

Fully Homomorphic Encryption

Instructor: Boaz Barak *Scribe: Alessandro Chiesa*

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 over 30 years. In 2009, Craig Gentry [Gen09a, Gen09b, Gen10] proposed the first fully-homomorphic encryption scheme that is secure under a reasonable assumption. Craig Gentry and Shai Halevi have already implemented the scheme, and are working on optimizations [GH10].

In the next two lectures, we will describe a somewhat simplified variant of Gentry's construction, obtained by Martin van Dijk, Craig Gentry, Shai Halevi and Vinod Vaikuntanathan [vDGHV10] (with a slight simplification suggested by Sushant Sachdeva, which was also independently observed by Ivan Damgård).

1 Definitions

Two sequences of distributions $X = \{X_n \in \Delta(\{0,1\}^{\mathrm{poly}(n)})\}_{n\in\mathbb{N}}$ and $Y = \{Y_n \in \Delta(\{0,1\}^{\mathrm{poly}(n)})\}_{n\in\mathbb{N}}$ are *statistically indistinguishable* if, for every function family $\{f_n \colon \{0,1\}^{\mathrm{poly}(n)} \to \{0,1\}\}_{n\in\mathbb{N}}$ and for all sufficiently large $n \in \mathbb{N}$, $\|\,\mathbb{E}[f(X_n)] - \mathbb{E}[f(Y_n)]\,\| < 2^{-n^{1/1000}}$, where $\|\cdot\|$ denotes the ℓ_1 norm.

The definition of *strongly* fully-homomorphic encryption is as follows:

Definition 1. We say that a quadruple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ of probabilistic polynomial-time algorithms is a *strongly fully-homomorphic (public-key) encryption scheme* (FHE for short) if:

1. $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a semantically-secure (public-key) encryption scheme[1] and,
2. For every polynomially-bounded function $t \colon \mathbb{N} \to \mathbb{N}$, every polynomial-size t-input circuit family $\{C_n\}_{n\in\mathbb{N}}$, every
3. For every sequence of key pairs $\{(\mathsf{pk}_n, \mathsf{sk}_n) \in \mathsf{Gen}(1^n)\}_{n\in\mathbb{N}}$, every polynomially-bounded function $t \colon \mathbb{N} \to \mathbb{N}$, every polynomial-size t-input circuit family $\{C_n\}_{n\in\mathbb{N}}$, every sequence of input bits $\{(b_{1,n}, b_{2,n}, \ldots, b_{t(n),n}) : b_{i,n} \in \{0,1\}\}_{n\in\mathbb{N}}$, every sequence of valid ciphertexts $\{(c_{1,n}, c_{2,n}, \ldots, c_{t(n),n}) : c_{i,n} \in \mathsf{Enc}_{\mathsf{pk}_n}(b_{i,n})\}_{n\in\mathbb{N}}$, we have:[2]
$$\left\{c^* \leftarrow \mathsf{Eval}_{\mathsf{pk}_n}(C_n, c_{1,n}, \ldots, c_{t(n),n}) \,:\, c^*\right\}_{n\in\mathbb{N}} \approx_s \left\{c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}_n}(C_n(c_{1,n}, \ldots, c_{t(n),n})) \,:\, c^*\right\}_{n\in\mathbb{N}} .$$

The second (very strong) condition implies, in particular, the following weaker one: for all sufficiently large $n \in \mathbb{N}$, for every $b_1, \ldots, b_{t(n)} \in \{0,1\}$, the following two distributions are statistically indistinguishable:

$$\big\{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b_1)\,;\, \ldots\,;\, c_{t(n)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b_{t(n)})\,;$$
$$c^* \leftarrow \mathsf{Eval}_{\mathsf{pk}}(C_n, c_1, \ldots, c_{t(n)}) \,:\, (\mathsf{pk}, c_1, \ldots, c_{t(n)}, c^*)\big\}_{n\in\mathbb{N}}$$

[1] That is, the two sequences of distributions given by $\{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0) \,:\, (\mathsf{pk}, b)\}_{n\in\mathbb{N}}$ and $\{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(1) \,:\, (\mathsf{pk}, b)\}_{n\in\mathbb{N}}$ are computationally indistinguishable, i.e., any circuit of polynomial size will succeed in distinguishing them with bias less than one over any polynomial; for the purposes of this class, we can replace "any polynomial" with some fixed super-polynomial function such as $2^{-n^{0.001}}$.

[2] We require *statistical* indistinguishability, because we want the two distributions to be close even for a distinguisher that knows the secret key sk. That will ensure that they decrypt to the same value!

and

$$\big\{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b_1)\,;\, \ldots\,;\, c_{t(n)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b_{t(n)})\,; \\ c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(C_n(c_1,\ldots,c_{t(n)}))\,:\,(\mathsf{pk}, c_1, \ldots, c_{t(n)}, c^*)\big\}_{n\in\mathbb{N}} \ .$$

This latter condition allows for Eval to fail on "bad" public keys or "bad" ciphertexts.

Recall that, in the definition of *weakly* fully-homomorphic encryption, the second condition is relaxed to an even weaker condition that only requires Eval to output ciphertexts not much longer than the input ciphertexts.

Our first observation is that it suffices to construct an evaluation algorithm Eval only for XOR and AND (or, equivalently, for addition and multiplication modulo 2), because $\{\mathrm{XOR}, \mathrm{AND}\}$ is a universal gate set and thus we can perform the evaluation of a circuit C gate by gate. (Though one still has to argue that, since the size of the circuit is much smaller than the statistical distance, performing an evaluation for each gate in the circuit will not increase the statistical distance by too much.) Hence, we will work with the following equivalent formulation of strongly fully-homomorphic encryption:

Definition 2. We say that a quadruple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$ of probabilistic polynomial-time algorithms is a *strongly fully-homomorphic (public-key) encryption scheme* (FHE for short) if:

1. $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a semantically-secure (public-key) encryption scheme and,
2. For every two bits b and b' in $\{0,1\}$:
   (a) for every three sequences $\{(\mathsf{pk}_n, \mathsf{sk}_n) \in \mathsf{Gen}(1^n)\}_{n\in\mathbb{N}}$, $\{c_n \in \mathsf{Enc}_{\mathsf{pk}_n}(b)\}$, and $\{c'_n \in \mathsf{Enc}_{\mathsf{pk}_n}(b')\}$,
   $$\big\{c^* \leftarrow \mathsf{Add}_{\mathsf{pk}_n}(c_n, c'_n)\,:\,c^*\big\}_{n\in\mathbb{N}} \approx_s \big\{c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}_n}(b \oplus b')\,:\,c^*\big\}_{n\in\mathbb{N}} \ .$$
   (b) for every three sequences $\{(\mathsf{pk}_n, \mathsf{sk}_n) \in \mathsf{Gen}(1^n)\}_{n\in\mathbb{N}}$, $\{c_n \in \mathsf{Enc}_{\mathsf{pk}_n}(b)\}$, and $\{c'_n \in \mathsf{Enc}_{\mathsf{pk}_n}(b')\}$,
   $$\big\{c^* \leftarrow \mathsf{Mult}_{\mathsf{pk}_n}(c_n, c'_n)\,:\,c^*\big\}_{n\in\mathbb{N}} \approx_s \big\{c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}_n}(b \oplus b')\,:\,c^*\big\}_{n\in\mathbb{N}} \ .$$

Similarly to Definition 1, the second condition implies, in particular, that

$$\big\{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b)\,;\, c' \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b')\,;\, c^* \leftarrow \mathsf{Add}_{\mathsf{pk}}(c, c')\,:\,(\mathsf{pk}, c, c', c^*)\big\}_{n\in\mathbb{N}} \\ \approx_s \big\{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b)\,;\, c' \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b')\,;\, c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b \oplus b')\,:\,(\mathsf{pk}, c, c', c^*)\big\}_{n\in\mathbb{N}} \ .$$

and

$$\big\{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b)\,;\, c' \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b')\,;\, c^* \leftarrow \mathsf{Mult}_{\mathsf{pk}}(c, c')\,:\,(\mathsf{pk}, c, c', c^*)\big\}_{n\in\mathbb{N}} \\ \approx_s \big\{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)\,;\, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b)\,;\, c' \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b')\,;\, c^* \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b \wedge b')\,:\,(\mathsf{pk}, c, c', c^*)\big\}_{n\in\mathbb{N}} \ .$$

However, for simplicity, we will use the stronger definition, as it is easier to work with (and we can anyways achieve it). Thus, henceforth we will focus on constructing a scheme satisfying Definition 2.

Private-key FHE. The definitions for private-key strongly fully-homomorphic encryption schemes are analogous, except that the algorithms Eval, Add, and Mult do not get the secret key as input. (Else, the "computing on ciphertext" task would be trivial: Eval could simply decrypt, evaluate the circuit on the plaintexts, and then re-encrypt.)

Also, one can easily transform a private-key FHE into a public-key FHE, by letting the public key be encryptions of zero and one. (In fact, this is an even easier transformation than the one of Rothblum [Rot10] that we discussed in the last lecture, and can be approached as an easier variant of the homework exercise.)

2 Construction of a "noisy" homomorphic encryption scheme

Our first step is to construct a primitive, which is weaker than a strongly fully-homomorphic encryption scheme, that we call a "noisy homomorphic encryption scheme". This primitive is an encryption scheme with Add and Mult algorithms that satisfies a relaxed notion of homomorphism. It will be easier to first construct the scheme, and only after define precisely the notion of homomorphism that it satisfies.

We remark that our notion of noisy homomorphism does 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

Let $N = PQ$, where P and Q are two large random primes, and suppose that we are given several random multiples of P, i.e., $R_1P \pmod N), \dots, R_tP \pmod N$ where the R_i's are chosen randomly in $\mathbb{Z}_Q$. Can we efficiently recover P from the t random products? In this case, recovering P can easily be done by computing the greatest common divisor of, say, R_1P and R_2P, because with good probability P will be the only common factor between them.

However, the algorithm for computing the greatest common divisor (which is the Euclidean algorithm) is extremely sensitive to noise (just like the algorithm for Gaussian elimination). In particular, it is not at all clear how to adapt the algorithm to (or even how to come up with a new efficient algorithm for) the case when we are given "noisy" random multiples of P, i.e., $R_1P + E_1 \pmod N, \dots, R_tP + E_t \pmod N$ where the E_i's are noise factors that are chosen in some interval $[-\mathbf{E}, +\mathbf{E}]$ for some noise parameter $\mathbf{E} \ll N$.

This motivates us to make the following computational assumption, on which we will base the security of our scheme:

Learning Divisor with Noise (LDN) assumption. Let P be a random n-bit prime, Q a random n^4-bit prime, set $N = PQ$, and choose any arbitrary $\mathbf{E} \geq 2^{n^{0.1}}$. Then, for any polynomially-bounded function $t\colon \mathbb{N} \to \mathbb{N}$ there is no polynomial-time algorithm that, on input $(N, \mathbf{E})$ and $X_1, \dots, X_t \in \mathbb{Z}_N$, can distinguish between the following two distributions:

1. the X_i's are independently drawn at random from $\mathbb{Z}_N$, and
2. the $X_i = PR_i + 2E_i \pmod N$, where R_i is drawn 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 lowercase letters (such as t) for values that are polynomial in the security parameter n, and capital letters (such as N, P, Q, and R) for large numbers that have poly(n) digits (and hence are exponential in the security parameter 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 will not 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 E_i) for convenience, but the two choices are equivalent. (Indeed, 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.)

5. The LDN assumption is clearly 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. (On the one hand, this is good news in light of Shor's factoring algorithm; on the other hand, it would have been fantastic to obtain FHE under more well-studied assumptions such as hardness of the RSA problem, factoring, or discrete logarithm.)

2.2 The noisy homomorphic encryption scheme

We now describe the noisy homomorphic encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$. The construction will yield a private key scheme (but that is not a problem, because it will be easy to convert it to the public key case). We will also assume some *public parameters*, which are values that are output by the key generation algorithm Gen and are made public. (Formally, such public parameters are never needed since we can always append them to the secret key and all ciphertexts, but it makes for cleaner notation to assume them.)

Key Generation: Draw a random n-bit prime P and a random n^4-bit prime Q. Set $N = PQ$, keep P as the secret key, and publish N as a public parameter.

Encryption: For $b \in \{0,1\}$, we let $\mathsf{Enc}_{N,P}(b) = \mathsf{Enc}_{N,P}^{2^{\sqrt{n}}}(b)$, where $\mathsf{Enc}_{N,P}^{\mathbf{E}}(b)$ is the distribution 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 $X - \lfloor X/P \rceil P \pmod 2$. (Where $\lfloor x \rceil$ denotes the integer closest to x, breaking ties, say, downwards.) In other words, find the nearest multiple of P to X, subtract it away to get the noise, and then the parity of the noise is the decrypted bit.

Security and correctness of scheme. The choice $2^{\sqrt{n}}$ for the parameter $\mathbf{E}$ makes the scheme both correct and secure. More concretely, the security follows through a simple hybrid argument: the LDN assumption implies that the following two distributions are computationally indistinguishable:

$$\big\{(N,P) \leftarrow \mathsf{Gen}(1^n)\,;\, X \leftarrow_{\mathrm{R}} \mathbb{Z}_N \,:\, (N, X)\big\} \text{ and}$$
$$\big\{(N,P) \leftarrow \mathsf{Gen}(1^n)\,;\, R \leftarrow_{\mathrm{R}} \mathbb{Z}_Q\,;\, E \leftarrow [-\mathbf{E}, +\mathbf{E}] \,:\, (N, RP + 2E \pmod{N})\big\}\ .$$

Hence, for $b \in \{0,1\}$,

$$\begin{aligned}
&\big\{(N,P) \leftarrow \mathsf{Gen}(1^n)\,;\, X \leftarrow_{\mathrm{R}} \mathbb{Z}_N \,:\, (N,X)\big\} \\
=& \big\{(N,P) \leftarrow \mathsf{Gen}(1^n)\,;\, X \leftarrow_{\mathrm{R}} \mathbb{Z}_N \,:\, (N,X+b)\big\} \\
\approx_c& \big\{(N,P) \leftarrow \mathsf{Gen}(1^n)\,;\, R \leftarrow_{\mathrm{R}} \mathbb{Z}_Q\,;\, E \leftarrow [-\mathbf{E},+\mathbf{E}] \,:\, (N, RP + 2E + b \pmod N)\big\} \ ,
\end{aligned}$$

as desired. 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 simply define the two operations Add and Mult as follows:

$$\begin{aligned}
\mathsf{Add}_N(X,X') &= X + X' \pmod N \text{ and} \\
\mathsf{Mult}_N(X,X') &= X \cdot X' \pmod N \ .
\end{aligned}$$

Let $\mathcal{E}^{\mathbf{E}}_{N,P}(b)$ denote the set of possible encryptions of b with parameter $\mathbf{E}$, public parameter N, and secret key P,

$$\mathcal{E}^{\mathbf{E}}_{N,P}(b) = \big\{X \,:\, X = RP + 2E + b \pmod N\,,\, R \in \mathbb{Z}_Q\,,\, E \in [-\mathbf{E},+\mathbf{E}]\big\} \ .$$

Note that if $\mathbf{E}' \geq \mathbf{E}$ then $\mathcal{E}^{\mathbf{E}}_{N,P}(b) \subseteq \mathcal{E}^{\mathbf{E}'}_{N,P}(b)$. Also, as long as $\mathbf{E}$ is not to close to P (say less than $P/10$) then the two sets $\mathcal{E}^{\mathbf{E}}_{N,P}(0)$ and $\mathcal{E}^{\mathbf{E}}_{N,P}(1)$ are disjoint.

If the scheme was fully homomorphic then we would have the following condition for $\mathbf{E} = 2^{\sqrt{n}}$: if $X \in \mathcal{E}^{\mathbf{E}}_{N,P}(b)$ and $X' \in \mathcal{E}^{\mathbf{E}}_{N,P}(b')$, then $\mathsf{Add}_N(X,X')$ is in $\mathcal{E}^{\mathbf{E}}_{N,P}(b \oplus b')$ (and moreover it is uniformly distributed in that set); similarly $\mathsf{Mult}_N(X,X')$ would be uniform in $\mathcal{E}^{\mathbf{E}}_{N,P}(b \wedge b')$. However, those conditions simply do not hold for our scheme.

To begin with, Add and Mult are not even randomized! Still, we can show that both Add and Mult do not add too much noise to the ciphertext:

Lemma 3. *For every $\mathbf{E}$ and $\mathbf{E}'$, if $X \in \mathcal{E}^{\mathbf{E}}_{N,P}(b)$ and $X' \in \mathcal{E}^{\mathbf{E}'}_{N,P}(b')$, then:*

- $\mathsf{Add}_N(X,X') \in \mathcal{E}^{\mathbf{E}+\mathbf{E}'}_{N,P}(b \oplus b')$, *and*
- $\mathsf{Mult}_N(X,X') \in \mathcal{E}^{3\mathbf{E}\mathbf{E}'}_{N,P}(b \wedge b')$.

Proof. Recall that $X = RP + 2E + b$ and $X' = R'P + 2E' + b'$. First, let us consider $\mathsf{Add}_N(X,X')$:

$$X + X' = (R + R')P + 2(E + E') + (b + b') \pmod N \ ,$$

and thus $X + X' \pmod N \in \mathcal{E}^{\mathbf{E}+\mathbf{E}'}_{N,P}$. Next, let us consider $\mathsf{Mult}_N(X,X')$:

$$X \cdot X' = KP + 2(2EE' + E + E') + b \cdot b' \pmod N \ ,$$

for some integer K and thus $X \cdot X' \pmod N \in \mathcal{E}^{3\mathbf{E}\mathbf{E}'}_{N,P}$. □

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 do not take a product of more than say $n^{1/10}$ of them then we will still get ciphertexts of noise much less than 2^n (and hence we can decrypt them). But how do we "keep track" of how noise grows in a possibly more complicated sequence of operations?

At high level, we relate the operations performed on the ciphertext to polynomials, and if the operations performed on the ciphertext have "low-degree", then we have some control on the noise of the ciphertext. Details follow.

Define an m-input *arithmetic circuit* to be a circuit C, taking m integers as input and producing one integer as output, that consists of only integer multiplication gates, integer addition gates, and the constants 0 and 1. Define $\mathcal{P}(C)$ to be the polynomial mapping from $\mathbb{Z}^m$ to $\mathbb{Z}$ that C computes. If we also think of C as a Boolean circuit (with integer multiplication gates as AND and integer addition gates as XOR) we get, for every $b_1, \dots, b_m \in \{0, 1\}$,

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

For a polynomial $f\colon \mathbb{Z}^m \to \mathbb{Z}$ and $M \geq 0$ we then define $|f|_M$ to be the maximum over all $x_1, \dots, x_m$ satisfying $|x_i| \leq M$ of $|f(x_1, \dots, x_m)|$. Then, if f is of degree d and all its monomials have coefficients of magnitude at most C, then $|f|_M \leq C \cdot m^d \cdot M^d$. (As can easily be seen by writing f as $\sum_{i_1+\cdots+i_m \leq d} \alpha_{i_1,\dots,i_m} x^{i_1} \cdots x^{i_m}$.)

We are now ready to formally characterize the extent to which our noisy homomorphic encryption scheme is indeed homomorphic:

Lemma 4. *Let C be an arithmetic circuit such that $|\mathcal{P}(C)|_{2\mathbf{E}+1} < P/10$. Then, for every $b_1, \dots, b_m \in \{0, 1\}$, if we let $X_i = \mathsf{Enc}_{N,P}(b_i)$ and evaluate the circuit C on $X_1, \dots, X_m$, using* Add *and* Mult *for the gates to obtain a result X^*, then we will have that*

$$\mathsf{Dec}_{N,P}(X^*) = C(b_1, \dots, b_m) \ .$$

Proof. We know that, for every $i = 1, \dots, m$, $X_i = R_i P + 2E_i + b_i$ 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

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

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

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

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

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

Now, by the same reasoning as above

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

for some integer K'' and hence

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

as desired. $\square$

3 Achieving full homomorphism

To convert a noisy homomorphic encryption scheme into a fully homomorphic one, we consider the solution approach of constructing the following two (additional) operations:

- Clean: takes as input the public parameter N and a ciphertext in $\mathcal{E}_{N,P}^{2^{n^{0.9}}}(b)$, and outputs a ciphertext in $\mathcal{E}_{N,P}^{2^{n^{0.3}}}(b)$. (I.e., it *reduces* the noise of the ciphertext.)

- ReRand: takes as input the public parameter N and a ciphertext in $\mathcal{E}_{N,P}^{2^{n^{0.4}}}(b)$, and outputs a ciphertext that is distributed statistically close to the right distribution over $\mathcal{E}_{N,P}^{2^{\sqrt{n}}}(b)$,[3] i.e., for every two sequences $\{(N_n, P_n) \in \mathsf{Gen}(1^n)\}_{n\in\mathbb{N}}$ and $\{X_n \in \mathsf{Enc}_{N_n,P_n}(b)\}_{n\in\mathbb{N}}$,

$$\left\{X' \leftarrow \mathsf{Enc}_{N_n,P_n}(b) \,:\, X'\right\}_{n\in\mathbb{N}} \approx_s \left\{X' \leftarrow \mathsf{ReRand}_{N_n}(X_n) \,:\, X'\right\}_{n\in\mathbb{N}} .$$

It is in fact easy to see how $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$, together with Clean and ReRand, implies a fully homomorphic encryption scheme: we simply modify the definition of Mult and Add to apply Clean and ReRand

- $\mathsf{Mult}_N(X, X') = \mathsf{ReRand}_N\big(\mathsf{Clean}_N(X \cdot X' \pmod{N})\big)$, and
- $\mathsf{Add}_N(X, X') = \mathsf{ReRand}_N\big(\mathsf{Clean}_N(X + X' \pmod{N})\big)$.

Now, given two ciphertexts X and X' encrypting b and b' respectively (i.e., $X \in \mathcal{E}_{N,P}^{2^{\sqrt{n}}}(b)$ and $X' \in \mathcal{E}_{N,P}^{2^{\sqrt{n}}}(b')$), the value $Y = X \cdot X' \pmod{N}$ will be in $\mathcal{E}_{N,P}^{3\cdot 2^{2\sqrt{n}}}(b \wedge b') \subseteq \mathcal{E}_{N,P}^{2^{n^{0.9}}}(b \wedge b')$, and hence $Z = \mathsf{Clean}(Y)$ will be in $\mathcal{E}_{N,P}^{2^{n^{0.3}}}(b \wedge b') \subseteq \mathcal{E}_{N,P}^{2^{n^{0.4}}}(b \wedge b')$, meaning that $\mathsf{ReRand}(Z)$ will be statistically indistinguishable from an encryption of $b \wedge b'$. A similar argument can be applied to Add.

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

Our goal is now to construct both Clean and ReRand, and Clean is really the important one among those (as the re-randomization property can usually be achieved for many encryption schemes).

Constructing ReRand. We briefly discuss how to obtain ReRand. (**Exercise: work out and verify the details!**) Recall that the input to ReRand is a ciphertext of the form $X = RP + 2E + b$ with $|E| \leq 2^{n^{0.4}}$; the goal is to transform X into $X' = R'P + 2E' + b$ where R' is uniform in $[Q] = [N/P]$ and E' is uniform in $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$. We distinguish between the two types of re-randomizations that we have to perform:

- *Re-randomizing the noise.* If we just wanted to re-randomize the noise E in X, we could simply draw E'' uniformly in $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$, and add $2E''$ to X. Then $E' = E + E''$ would be distributed uniformly in the interval $[-2^{\sqrt{n}}, +2^{\sqrt{n}}] + E$ which is within $2^{n^{0.4}}/2^{\sqrt{n}} = \mathsf{negl}(n)$ statistical distance to the uniform distribution over $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$.

[3]Similarly to the property for Eval, this property for ReRand implies a weaker one that suffices for most applications, namely, that re-randomization is allowed to fail on "bad" keys or "bad" ciphertexts:

$$\big\{(N, P) \leftarrow \mathsf{Gen}(1^n)\,;\, X \leftarrow \mathsf{Enc}_{N,P}(b)\,;\, X' \leftarrow \mathsf{Enc}_{N,P}(b) \,:\, (N, X, X')\big\}_{n\in\mathbb{N}}$$
$$\approx_s \big\{(N, P) \leftarrow \mathsf{Gen}(1^n)\,;\, X \leftarrow \mathsf{Enc}_{N,P}(b)\,,\, X' \leftarrow \mathsf{ReRand}_N(X) \,:\, (N, X, X')\big\}_{n\in\mathbb{N}} .$$

- *Re-randomizing the multiple.* Re-randomizing the multiple R in X is a bit more tricky. The high-level idea is as follows. Suppose that we have at our disposal many, say $X_1, \ldots, X_m$ for $m = n^6$, random encryptions of 0 with small noise (e.g., less than $2^{n^{0.4}}$). Then we will choose a *random subset* $S \subseteq [m]$ and consider the ciphertext $X'' = X + \sum_{i \in S} X_i$; this is still an encryption of 0, with at most $(m+1)2^{n^{0.4}}$ noise, and the corresponding multiple is just

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

  where $X_i = PR_i + 2E_i$. The goal is to show that the above expression is close in statistical distance to the uniform distribution over $\mathbb{Z}_Q$. We use the following lemma, which is a variant of what is known as the "leftover hash lemma":

  Lemma 5. *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 $R_1, \ldots, R_m$, 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 $\mathbb{Z}_Q$.*

  In other words, the randomness comes from the *choice of S*; intuitively, the lemma is reasonable because $2^m = 2^{n^6} \gg Q \approx 2^{n^4}$.

  You can prove the lemma in a way similar to 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 at random the $R_1, \ldots, R_m$ 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 $\mathbb{E}[X_S] = 1/Q$, $\mathbb{E}[X_S X_T] = 1/Q^2$, and then 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 $R_i \leftarrow_{\mathrm{R}} [Q]$ and $E_i \leftarrow_{\mathrm{R}} [-2^{n^{0.4}}, +2^{n^{0.4}}]$. Then to re-randomize X we choose a random subset S of $[m]$, and E'' at random from $[-2^{\sqrt{n}}, +2^{\sqrt{n}}]$ and output

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

Constructing Clean with "wishful thinking". We now tackle the bigger problem: how to obtain the clean-up procedure Clean? Obtaining Clean appears to be very challenging, because, up until this point, any operation that we performed on ciphertexts (such as adding, multiplying, or re-randomizing) only *increased* the noise. In fact, it seems somewhat counterintuitive that you could decrease the noise of a ciphertext without knowing the secret key, since if you could decrease it *too much* then you would be able to figure out the corresponding plaintext!

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

Specifically, consider the decryption algorithm Dec: it takes as input the secret key P and a ciphertext X, and outputs the corresponding bit b. Since P and X are themselves just strings of bits, Dec is a function mapping $\{0,1\}^m$ to $\{0,1\}$, where $m = |P| + |X| = n + n^5$ is the length of this description. (Note that this m is not the same number m that we used in the ReRand operation.)

More than that, the decryption algorithm Dec is an *efficient* function, and so it can be computed by a polynomial-size Boolean circuit C, and we can assume (without loss of generality) that the gates of this circuit only consist of AND and XOR gates (as well as the constants 0 and 1) since the two gates form a universal set.

Now suppose that we are lucky: it is the case that $|\mathcal{P}(C)|_{2^{n^{0.1}}} < 2^{n^{0.3}}$. For example, this could 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 perform the Clean operation. Details follow.

Recall that Clean is given as input $X = RP + 2E + b$ where $|E| \leq 2^{n^{0.9}}$ and it should output $X' = R'P + 2E' + b$ such that $|E'| \leq 2^{n^{0.3}}$. We first modify the scheme to include $Y_1, \ldots, Y_n$ in the public parameters where $Y_i = \mathsf{Enc}_{N,P}^{2^{n^{0.1}}}(P_i)$ and P_i is the i-th bit of the secret key P; i.e., we include an encryption of P in the public parameters, using a noise value $2^{n^{0.1}}$ which is smaller than the standard parameter, but still big enough to ensure security. The procedure Clean is then defined as follows:

1. We take $Y_1, \ldots, Y_n$ from the public parameters; then 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, and so on.) 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.

   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! (Each one of them has noise at most $2^{n^{0.1}}$.)

2. 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 last thing to check is that the noise of X' is not too large. And this follows from the fact that the circuit is simple and from the fact that we started from ciphertexts with small noise. Specifically, the noise level we will 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, the definition of CPA security (i.e., semantic security) does *not* guarantee that it is secure to encrypt the secret key. (**Exercise: give a counterexample!**) We overcome this issue by using a common cryptographic technique— making an assumption: we will assume that, even given oracle access to the encryption algorithm, it is computationally difficult to distinguish between an encryption of the secret key and an encryption of the all-zero string.

The requirement captured by our assumption is called *circular security*, and it is a specific case of the 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 (such as El-Gamal) 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 to even satisfy some notions of KDM security) under relatively standard assumptions. (E.g., see [BHHO08].) 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 to be circular secure.

In any case, we will assume that this scheme is circular secure. Hopefully at some point someone will manage to prove that it is, and get rid of this assumption.[4]

Again, the problem of constructing *any plausible* homomorphic encryption scheme, even with only heuristic security (such as in the Random Oracle Model), was open for 30 years, so we should not complain too much even if the solution uses somewhat non-standard assumptions.

A major issue. The major issue, however, is that we have no reason to believe that the decryption circuit has small norm! Generally, a circuit over $\{0,1\}^m$ is expected to have polynomial degree in $m \approx n^5$, and indeed it seems that one could *verify* that the decryption circuit actually computes a polynomial of degree at least $n/100$, and will satisfy $|\mathcal{P}(C)|_1 > 2^{n/100}$. So we are off by a polynomial factor in the exponent.

Next time 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.

References

[BHHO08] Dan Boneh, Shai Halevi, Mike Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In *Proceedings of the 28th Annual International Cryptology Conference*, CRYPTO '08, pages 108–125, Berlin, Heidelberg, 2008. Springer-Verlag. `http://crypto.stanford.edu/~dabo/abstracts/circular.html`.

[Gen09a] Craig Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford University, 2009. `http://crypto.stanford.edu/craig`.

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.

[Gen10] Craig Gentry. Computing arbitrary functions of encrypted data. *Communications of the ACM*, 53:97–105, March 2010.

[GH10] Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520, 2010. `http://eprint.iacr.org/2010/520`.

[Rot10] Ron Rothblum. Homomorphic encryption: from private-key to public-key. Electronic Colloquium on Computational Complexity, TR-10-146, 2010. `http://www.eccc.uni-trier.de/report/2010/146/`.

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '10, pages 24–43, Berlin, Heidelberg, 2010. Springer-Verlag.

[4]Even without this assumption one can get a *limited* homomorphic encryption scheme, where the public key grows with the depth of the circuit one wishes to evaluate. See Gentry's thesis [Gen09a].