6.889 New Developments in Cryptography February 1, 2011

Introduction to Homomorphic Encryption

Instructors: Shafi Goldwasser, Yael Kalai, Leo Reyzin, Boaz Barak, and Salil Vadhan

Lecture by Shafi Goldwasser.

Scribed by Rachel Miller.

1 Administrative Details

Instructors for the course are Shafi Goldwasser (MIT), Yael Kalai & Boaz Barak (Microsoft Research), Leo Reyzin (BU), and Salil Vadhan (Harvard).

Classes will meet on Tuesdays at 12:30-2:30 with a short break in between, and some Friday afternoons.

Student work for the course will be to attend lectures, and to scribe a lecture. People may team up to scribe lectures.

Pre-requisites for the course are 6.046, and 6.875. Students are welcome to try to take this course concurrently with 6.875, but it will be harder!

Tentative Unit List includes:

1. Computation on Encrypted Data - current constructions of Homomorphic Encryption, multi-party computation techniques. (Includes 2 lectures by Shafi, 3 by Boaz).
2. Leakage & Side Channel Attack Resilience (with lectures by Yael, Shafi and Leo)
3. Techniques from Extractors - and how these techniques apply to work in leakage (lectures by Leo)
4. Differential Privacy - what is it and can we achieve it (lectures by Salil)

A tentative schedule of lectures was passed out, and will be put on a website.

2 Preliminaries

We use PPT to denote the class of algorithms that are in probabilistic polynomial time. For PPT algorithm A on input x, we let $A(x)$ denote the probability distribution defined over the output of A induced by the choice of x and A's coin tosses.

We will use the notation $x \xleftarrow{\$} S$ to mean that x is chosen with uniform probability in set S.

2.1 Statistical and Computation Closeness

Let $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$ be sequences of probability distributions.

We say $\mathcal{X}$ and $\mathcal{Y}$ are *perfectly indistinguishable* if $\exists n_c$ such that for all $n > n_c$, $X_n = Y_n$.

We say $\mathcal{X}$ and $\mathcal{Y}$ are *statistically indistinguishable*, or statistically close, if $\forall c > 0$, $\exists n_c$ such that $\forall n > n_c$,

$$\Delta(X_n, Y_n) := \frac{1}{2} \sum_{\sigma\in\{0,1\}^n} |\Pr[X_n = \sigma] - \Pr[Y_n = \sigma]| < \frac{1}{n^c}.$$

In other words, the sum of the difference in probabilities of elements occurring, must become negligible in n. We call the $\Delta(X_n, Y_n)$ the *statistical distance* between X_n and Y_n. Notationally, we say $\mathcal{X}$ and $\mathcal{Y}$ are statistically indistinguishable by $\mathcal{X} \approx_s \mathcal{Y}$.

Computational indistinguishability instead requires that an adversary with polynomial running time must be able to guess which distribution it has access to, with only negligible probability. More precisely, we say $\mathcal{X}$ and $\mathcal{Y}$ are *computationally indistinguishable* if $\forall D \in \mathsf{PPT}$, $\forall c > 0$ $\exists n_c$ such that $\forall n > n_c$,

$$|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| < \frac{1}{n^c}.$$

Notationally, we say $\mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable by $\mathcal{X} \approx_c \mathcal{Y}$.

2.2 Number Theory notes written by Pablo Azar, and edited by me

One useful value in groups is the *Legendre Symbol*: when p is an odd prime, the Legendre symbol is denoted by $\left(\frac{a}{p}\right)$, and is defined

$$\left(\frac{a}{p}\right) := a^{\frac{p-1}{2}}.$$

The *Jacobi Symbol* is a generalization of Legendre Symbol to odd composites n. In particular, if $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$, the Jacobi symbol is defined as

$$\left(\frac{a}{n}\right) := \Pi_{i=1}^{k} \left(\frac{a}{p_i}\right)^{\alpha_i}.$$

Both the Legendre and Jacobi Symbols are efficiently computable.

We say that a is a *quadratic residue* modulo p if there exists $x \in \mathbb{Z}_p^*$ such that $a \equiv x^2 \bmod p$. We use QR_n to denote the group of quadratic residues modulo n. It's efficiently computable to determine whether an element is in QP_p for a prime p using Euler's criterion: the element a is a square modulo p if and only if $a^{\frac{p-1}{2}} \equiv 1 \bmod p$. Moreover, a is not a square modulo p iff $a^{\frac{p-1}{2}} \equiv -1 \bmod p$.

Let integer $N = p \cdot q$, where p and q are primes of equal length satisfying $p \equiv q \equiv 3 \mod 4$. (Such N are called Blum integers.) We use $\mathbb{Z}_N^*$ to denote the group of integers in $\mathbb{Z}_N$ relatively prime to N, $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N : \gcd(x, N) = 1\}$. We use $\mathbb{J}_N$ to the elements in $\mathbb{Z}_N^*$ with Jacobi Symbol $(+1)$.

Can we efficiently determine whether a is a square modulo this composite N? Given the factorization of N, this can be done efficiently- using the Chinese Remainder Theorem, its clear to see a is a QR modulo N iff its a QR modulo both p and q, which can be checked efficiently. However, without the factorization of N, this is not thought to be possible. This is formalized by the *quadratic residuosity* (QR) assumption, which assumes that the uniform distributions over $\mathbb{J}_N$ and $\mathbb{QR}_N$ are computationally indistinguishable when the factorization of N is not known.

In order for a to be a square modulo N, it needs to be a square modulo all prime factors of N. This implies for $N = p \cdot q$, if the Jacobi symbol $\left(\frac{a}{N}\right)$ is equal to -1, a is not a square. However, if the Jacobi symbol $\left(\frac{a}{N}\right)$ is equal to $+1$, it gives no information about whether a is a square or not.

The *Euler totient function* $\phi(n)$ is defined by $\phi(n) := |\mathbb{Z}_n^*|$. Equivalently, $\phi(n)$ is the number of integers between 1 and n that are relatively prime to n.

2.3 Leftover Hash Lemma

A great reference for the Leftover Hash Lemma is Oded Goldreich's book *Computational complexity: a conceptual perspective*, in Appendix D. The material in this subsection is taken from there.

Hashing is used extensively in cryptography to map large sets into smaller sets in a deterministic manner; by randomly selected a hash function from a family of hash functions, we hope that this mapping will be close to the uniform distribution in some way.

First, we describe a useful property a family of functions can have.

Definition 1 (t-wise Independence) *A family of functions $\{H_n^m\}_{m<n}$ from n-bit strings to m-bit strings is called* t-wise independent *if for every t distinct domain elements $x_1, \ldots, x_t \in \{0,1\}^n$ and every $y_1, \ldots, y_t \in \{0,1\}^m$ it holds that*

$$\Pr\left[h \xleftarrow{\$} H_n^m; \wedge_{i=1}^t h(x_i) = y_i\right] = 2^{-t \cdot m}.$$

We generally refer to 2-wise independent families of functions as *pairwise* independent.

A useful lemma concerning the "almost uniform" coverage of pair-wise independent hash functions is the following.

Lemma 2 *Let $m \leq n$ be integers, H_n^m be a family of pairwise independent hash functions, and $S \subseteq \{0,1\}^n$. Then, for every $y \in \{0,1\}^m$ and every $\epsilon > 0$, for all but at most a $\frac{2^m}{\epsilon^2 |S|}$*

fraction of $h \in H_n^m$ *it holds that*

$$(1-\epsilon)\cdot\frac{|S|}{2^m} < |\{x \in S : h(x) = y\}| < (1+\epsilon)\cdot\frac{|S|}{2^m}.$$

Finally, we give the Leftover Hash Lemma, which has many uses in cryptography.

Lemma 3 (Leftover Hash Lemma) *Let* $m \leq n$ *be integers,* H_n^m *be a family of pair-wise independent hash functions, and* $S \subseteq \{0,1\}^n$*. Consider random variables* X *and* H *that are uniformly distributed on* S *and* H_n^m *respectively. Then the statistical difference between* $(H, H(X))$ *and* (H, U_m) *is at most* $2\sqrt{\frac{2^m}{|S|}}$*.*

3 Homomorphic Encryption

We begin by defining semantically secure encryption schemes, then give a high level description of what additional properties homomorphic encryption (HE) schemes have. After further motivating HE schemes, we will give a formal definition of Homomorphic Encryption. Last, we will examine known constructions of "partially homomorphic" encryption schemes.

3.1 Encryption Schemes

In its most basic form encryption schemes are comprised of a three tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ - key generation, encryption, and decryption respectively.

In a public key encryption scheme with security parameter k, the key generation algorithm Gen takes as input the security parameter 1^k , and outputs a public-key, secret-key pair (pk, sk). The encryption algorithm E takes a public key, a message from a message space $\mathcal{M}$ and outputs a ciphertext C. For simplicity, we'll assume $\mathcal{M} = \{0,1\}^k$. Finally, the decryption algorithm D takes a secret-key sk and a ciphertext C and outputs a message M.

We have two requirements for public-key encryption schemes:

- **Correctness**: $\forall k, \forall M \in \{0,1\}^k, \Pr\left[(pk, sk) \leftarrow \mathsf{Gen}(1^k) : \mathsf{D}(sk, \mathsf{E}(pk, M)) = M\right] = 1.$
- **Semantic Security**: this essentially guarantees a PPT adversary can't gain any partial information from a ciphertext (and the public key used to generate it) about the message it encrypts.

  A scheme is said to be *semantically secure* if $\forall \mathcal{A} \in \mathsf{PPT}$, $\forall c > 0$, $\exists K$ such that $\forall k > K$, the probability that $\mathcal{A}$ "wins" the following game is less than $\frac{1}{2} + \frac{1}{k^c}$.

  - $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$
  - $\mathcal{A}$ is given pk, and produces $m_0, m_1 \in M_k$

- $b \xleftarrow{\$} \{0,1\}$, and $\mathcal{A}$ is given a ciphertext C generated as $C \xleftarrow{\$} \mathsf{E}(pk, m_b)$
- $\mathcal{A}$ outputs $b' \in \{0,1\}$, and is said to *win* if $b' = b$

In other words, for any polynomial time adversary $\mathcal{A}$ given a randomly generated public-key, it cannot find two messages m_0 and m_1 for which it can guess whether a ciphertext encrypts m_0 or m_1, except with negligible advantage over $\frac{1}{2}$.

In a private-key encryption scheme (also called symmetric-key encryption schemes), the secret key and the public key are the same. Put differently, the Gen algorithm generates only one key.

The semantic security requirement changes slightly for a private-key encryption scheme:

- **Private Key encryption Semantic Security** : we say a private-key scheme satisfies semantic security if $\forall \mathcal{A} \in \mathsf{PPT}$, $\forall c > 0$, $\exists K$ such that $\forall k > K$, the probability that $\mathcal{A}$ "wins" the following game is less than $\frac{1}{2} + \frac{1}{k^c}$.
  - $sk \leftarrow \mathsf{Gen}(1^k)$
  - $\mathcal{A}$ on input 1^k produces two messages $m_0, m_1 \in \{0,1\}^k$
  - $b \xleftarrow{\$} \{0,1\}$, and $\mathcal{A}$ is given a ciphertext $c_b \xleftarrow{\$} \mathsf{E}(sk, m_b)$
  - $\mathcal{A}$ outputs $b' \in \{0,1\}$, and is said to *win* if $b' = b$

We will later use public and private-key schemes with *multiple message indistinguishability*; here, instead of two messages, $\mathcal{A}$ produces two tuples of messages $(m_1^0, \ldots, m_t^0)$ and $(m_1^1, \ldots, m_t^1)$, and then is given a tuple of ciphertexts $(c_1, \ldots, c_t)$ generated as $c_i \xleftarrow{\$} \mathsf{E}(sk, m_i^b)$. $\mathcal{A}$ then is required to guess which tuple it received an encryption of. Unlike the public-key setting, multiple message indistinguishability does not follow immediately from single message indistinguishability.

3.2 High Level Description of Homomorphic Encryption

Now that we've described the basics of encryption scheme, we extend the notion to *Homomorphic Encryption* (HE) schemes. In addition to the standard requirements for encryption schemes, we'd like a HE scheme to also have a special associated PPT Eval algorithm; Eval should take ciphertexts as input, and then output a new ciphertext which decrypts to a pre-specified function of the plaintexts underlying the input ciphertexts. In particular, Eval will take a description of a function t-input function f and t ciphertexts generated as $C_1 \xleftarrow{\$} \mathsf{E}(M_1), \ldots, C_t \xleftarrow{\$} \mathsf{E}(M_t)$, and output a special ciphertext C^* that decrypts to $f(M_1, \ldots, M_t)$. We'd like C^* to hide all information about $M_1, \ldots, M_t$ and about $f(M_1, \ldots, M_t)$. An additional wish would be for C^* not to yield information about $M_1, \ldots, M_t$ even to the legal owner of the secret key.

The description of f could be a TM, a branching-program, a formula, a polynomial, or whatever. Most generally, it is specified by a polynomial size Boolean circuit, which we will assume here.

3.3 Motivations for Homomorphic Encryption

Delegation

It would be useful for a computationally weak "client" to be able delegate the processing of data to some more powerful third party "server", while still maintaining data privacy. To this end, the client could send the server an encryption of the data, created employing an HE scheme. The server is able to run processes over the encrypted data, and return an output to the client; the client needs only to decrypt to receive the processed answer. The server here can actually be a collection of computing devices, ie, "the cloud". As an example of a client/server relationship, a client smart card might want to gain access to CSAIL but be too computationally weak to answer the door's challenge question; by outsourcing this computation to some server while maintaining the privacy of its secret, the client smart card can hope to gain access to the building.

A variant of HE forbids the ciphertext C^* the client receives from computationally revealing any information about the process the server runs on the data. In a motivating example, consider x to be the private medical data of a consumer looking to buy medical insurance. An insurance company has some complicated policy P for determining the price of a premium for a patient based on their medical information. The potential client wants x to remain private, and so sends an encryption to the insurance company; the insurance company runs P on the encrypted data, and returns an encryption of that clients potential premium. Here, the insurance company wants insurance that the client gains no information about P- P is very secret proprietary information in this case!

Remote File Storage

Data storage in remote locations (ie, the cloud) is becoming increasingly common, but privacy is a big concern here. Client outsourcing of data processing to the cloud is becoming increasingly common, too. HE allows for the union of the seemingly incongruous goals of privacy for the remote data, while also allowing the server to give useful information about the data.

In a motivating example, consider a user that wants to run a keyword search on its entire set of encrypted data. Without HE, since the server can't tell which documents contain the keyword, it would be forced to send the entire set of encrypted data back to the user, who could decrypt it and look for the keyword. *With* HE, however, the server can simply run the keyword search algorithm with the encrypted keyword and the set of encrypted data, and send an encrypted list of documents containing the keyword back to the user.

Software Protection

Software protection is a huge issue for software companies- they put a lot of effort into writing code, but then others can easily examine the code to learn all its tricks and secrets.

It would be nice if the software companies could instead sell encryptions of their software, which reveal no meaningful information about code.

To this end, the company could instead sell a HE of the code. The user would encrypt their inputs to the program using HE. Both of these are sent as inputs to a universal circuit evaluator- which takes a description of a circuit (the proprietary software) in addition to the circuits inputs (the user's input to the software), and outputs the result of the circuit applied to the inputs. The output, then, is an encryption of the output of the original program. If the software company can require the user to be online to find out the program output, the software company can decrypt this value and send the output back to the user. Here, by giving the encrypted software to the user, the user still runs most of the computation using their own resources, even though the vendor must do computation for this decryption. A similar assumption requires the user to have secure tamper-free hardware to decrypt the final output.

3.4 Formal Definition of Fully Homomorphic Encryption

Definition 4 *A* Homomorphic Encryption scheme *is comprised of a tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{E}, \mathsf{D}, \mathsf{Eval})$*, and is defined with respect to a circuit* $\boldsymbol{C}$ *with* t *inputs. Though a HE scheme can be either a public-key or symmetric-key system, we will define it as a public-key system here.*

The key generation algorithm Gen *takes the security parameter* 1^k *as input, and outputs the public key and private key for the system (Notation:* $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$*). For simplicity, we'll assume that messages* $M \in \{0,1\}^{\ell(k)}$*. The encryption algorithm* E *takes a public key and a message as input, and outputs a ciphertext* C*, (Notation:* $C \leftarrow \mathsf{E}(pk, M)$ *for* $M \in \{0,1\}^{\ell(k)}$*). The decryption algorithm* D *takes a secret key and a ciphertext, and returns a message, (Notation:* $M \leftarrow \mathsf{D}(sk, C)$ *and* $M \in \{0,1\}^{\ell}$*). Finally, the homomorphic evaluation algorithm* Eval *takes as input a public key, a description of a* t*-input circuit* $\boldsymbol{C}$*, and* t *ciphertexts* $C_1, \ldots, C_t$ *such that* $C_i \xleftarrow{\$} \mathsf{E}(pk, M_i)$*, and produces as output* C^**, (Notation:* $C^* \leftarrow \mathsf{Eval}(pk, \boldsymbol{C}, C_1, \ldots, C_t)$*).*

We add a new correctness property to the standard correctness requirement for an encryption scheme as follows. We say that an encryption scheme is homomorphic with respect to *a* t*-input circuit* $\boldsymbol{C}$ *if* $\forall k$, $\forall M_1, \ldots, M_t$,

$$\Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^k); C_1, \ldots C_t \leftarrow \mathsf{E}(pk, M_1), \ldots, \mathsf{E}(pk, M_t); C^* \leftarrow \mathsf{Eval}(pk, \boldsymbol{C}, C_1, \ldots, C_t) :$$
$$\mathsf{D}(sk, C^*) = \boldsymbol{C}(M_1, \ldots, M_t)] = 1.$$

Similarly, a scheme is homomorphic with respect to a family of circuits $\{\mathbf{C}_i\}$ if the correctness property holds for any circuit $\mathbf{C} \in \{\mathbf{C}_i\}$.

Note that so far, our definition makes no requirement that the output C^* of Eval should look like a standard ciphertext. Indeed, without some additional restriction on C^*, every standard encryption scheme $(\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ can be trivially modified to yield a homomorphic

encryption scheme $(\mathsf{Gen}', \mathsf{E}', \mathsf{D}', \mathsf{Eval}')$ with respect to all circuits as follows.

- Gen' runs as Gen.
- E' runs as E.
- The Eval' is constructed to take a public key, a circuit description, and up to t ciphertexts, and then output the circuit description concatenated with each of the ciphertexts, as $C^* \leftarrow \mathsf{Eval}'(pk, \mathbf{C}, C_1, \ldots, C_t) = \mathbf{C}|C_1|\ldots|C_t$, with $|$ used to denote concatenation.
- On special ciphertexts C^* containing a circuit description, D' parses its input into $\mathbf{C}, C_1, \ldots, C_t$, runs the original decryption algorithm D on the ciphertexts to obtain messages $M_i \leftarrow \mathsf{D}(sk, C_i)$, and runs the circuit $\mathbf{C}$ on these messages, to obtain $\mathsf{D}'(sk, C^*) = \mathbf{C}(M_1, \ldots, M_t)$, satisfying the homomorphic correctness property. On ciphertexts without circuit descriptions, $\mathsf{D}'(sk, C)$ simply returns $\mathsf{D}(sk, C)$.

This HE scheme extension of basic encryption always exist, but is of little use for most applications. In order to make the problem non-trivial and to add interesting functionalities, we need to make additional restrictions on the output of Eval.

We consider two such additional restrictions:

1. **Strongly Homomorphic Encryption:** this restricts the output of the Eval function C^* to have the same distribution as a normal ciphertext encrypting the output of the circuit. Formalized, for HE over a t-input circuit $\mathbf{C}$, $\forall k$, $\forall M_1, \ldots, M_t$, when $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, $C_1, \ldots, C_t \leftarrow \mathsf{E}(pk, M_1), \ldots \mathsf{E}(pk, M_t)$, then the following two probability distributions are "the same"

   $$(\mathsf{Eval}(pk, \mathbf{C}, C_1, \ldots, C_t), C_1, \ldots, C_t)$$

   and

   $$(\mathsf{E}(\mathbf{C}(M_1, \ldots, M_t), C_1, \ldots, C_t)\,.$$

   Variations of strongly homomorphic encryptions do not require these two distributions to be the same but merely statistically-close or even only computationally-close.

2. **Weakly Homomorphic Encryption or compactness:** instead of requiring the outputs of E and Eval to be the same type of ciphertext, we simply require that the size ciphertext produced by Eval is independent of t, the number of inputs to circuit $\mathbf{C}$. Note that since the size of $\mathbf{C}$ grows at least as quickly as the number of inputs t, this also implies that the size of the produced ciphertext is independent of the size of $\mathbf{C}$. This requires "compactness", where $\exists$ a fixed polynomial bound $B(\cdot)$ st for $\forall k$ sufficiently large, $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, $\forall M_1, \ldots, M_t \in \{0,1\}^\ell$, $C_1, \ldots, C_t \leftarrow \mathsf{E}(pk, M_1), \ldots, \mathsf{E}(pk, M_t)$, $|\mathsf{Eval}(pk, C_1, \ldots, C_t)| < B(k)$.

As mentioned in the discussion of delegation, an additional restriction that can be considered prevents the output of Eval from revealing any information about the description of $\mathbf{C}$ itself.

Through this description, we've been discussing homomorphism with respect to specific circuit $\mathbf{C}$. A homomorphic encryption scheme with respect to all polynomial size Boolean circuits is called a *Fully Homomorphic Encryption scheme* (FHE scheme). It was long debated whether FHE schemes existed; the first FHE scheme was introduced by Craig Gentry in 2009 using assumptions on ideal lattices. FHE schemes now also were shown to exist under an assumption called approximate integer gcd. Existing FHE schemes are currently very inefficient- but present a great promise for the future. Since FHE are currently so inefficient and known under non-standard assumptions, encryption schemes that are homomorphic for smaller classes of circuits are still quite interesting, and easier to approach than FHE.

Remark: the evaluation algorithm is only required to work on ciphertexts created directly by encrypting messages; in particular, it may not work on the when run on special ciphertexts C^* that are outputs of the evaluation algorithm.

3.5 Examples of Homomorphic Encryption

RSA

The RSA encryption scheme was first publicly described by Rivest, Shamir and Adleman in 1978. RSA, as with other encryption schemes, is actually a triple of PPT algorithms $(\mathsf{Gen}_{\mathsf{RSA}}, \mathsf{E}_{\mathsf{RSA}}, \mathsf{D}_{\mathsf{RSA}})$. For security parameter k, the scheme works as follows.

- The key generation $\mathsf{Gen}_{\mathsf{RSA}}$ takes 1^k as input, randomly chooses two k-bit primes p and q, and computes $N = p \cdot q$. After choosing e uniformly at random from $\mathbb{Z}^*_{\phi(N)}$, it generates $d \equiv e^{-1} \mod \phi(N)$ - note $\phi(N) = (p-1) \cdot (q-1)$. Then $\mathsf{Gen}_{\mathsf{RSA}}$ outputs $((n, e), d)$ as its (pk, sk) pair.
- Encryption of a message $M \in \mathbb{Z}^*_N$ yields $\mathsf{E}((N, e), M) \equiv M^e \mod N$.
- Decryption of a ciphertext $C \in \mathbb{Z}^*_N$ returns $\mathsf{D}(d, C) \equiv C^d \mod N$.

Decryption works properly here because for $C = \mathsf{E}((N, e), M) \equiv M^e \mod N$, $\mathsf{D}(d, C) \equiv (M^e \mod N)^d \mod N \equiv M^(de) \mod N$. Since $d \cdot e \equiv 1 \mod \phi(N)$, there exists an integer k such that $\mathsf{D}(d, C) \equiv M^(k\phi(N) + 1) \mod N \equiv M \cdot \left(M^{\phi}(N)\right)^k \mod N \equiv M \mod N$. (This uses Euler's Theorem, which says for any $a \in \mathbb{Z}^*_n, a^{\phi(n)} \equiv 1 \bmod n$.)

This system is easily shown to be strongly homomorphic with respect to multiplication modulo N. Let $M_1, \ldots, M_t \in \mathbb{Z}^*_N$ with corresponding ciphertexts $C_1 = \mathsf{E}((N, e), M_1), \ldots, C_t = \mathsf{E}((N, e), M_t) \equiv M_t^e \mod N$, and let $\mathbf{C}$ describe multiplication of t messages. Then we can define $\mathsf{Eval}((N, e), \mathbf{C}, C_1, \ldots, C_t) \equiv C_1 \cdot \ldots \cdot C_t \mod N \equiv M_1^e \cdot \ldots \cdot M_t^e \equiv (M_1 \cdot \ldots \cdot M_t)^e$, giving the desired result.

Unfortunately, RSA as described here lacks semantic security- and we want better for our HE schemes. (In fact, any scheme with a deterministic encryption algorithm *can't* be semantically secure- an adversary can simply encrypt m_0 and m_1 themselves, and check which matches the challenge ciphertext.)

Today we use a modified randomized version of RSA. Lets consider a specific modification to RSA:

- To encrypt $x \in \{0,1\}$, choose $r \xleftarrow{\$} \mathbb{Z}_N^*$, and send $C = (r^e \mod N, msb(r) \oplus x)$, where msb denotes the most significant bit of r modulo N.

Interesting question: can we do something clever here to give homomorphic encryption using this scheme over some interesting family of circuits?

Quadratic Residuosity

A second scheme with homomorphism is the Quadratic Residuosity (QR) scheme, developed by Goldwasser and Micali in 1982.

The system utilizes a Blum Integer $N = p \cdot q$ as a public key (with primes $p \equiv q \equiv 3 \mod 4$ of appropriate size), with the factorization of N as the private key. The system encrypts messages $M \in \{0,1\}$. To generate $\mathsf{E}(0)$, select $r \xleftarrow{\$} \mathbb{Z}_N^*$, and let the ciphertext $C \equiv r^2 \mod N$. To generate $\mathsf{E}(1)$, select $r \xleftarrow{\$} \mathbb{Z}_N^*$, and let the ciphertext $C \equiv -r^2 \mod N$. Note that the distribution of $\mathsf{E}(0)$ is uniform over QR_N, while together the distributions of $\mathsf{E}(0)$ and $\mathsf{E}(1)$ are uniform over elements with Jacobi symbol of $(+1)$. By the *quadratic residuosity* (QR) assumption is that the uniform distributions over $\mathbb{J}_N$ and $\mathbb{QR}_N$ are computationally indistinguishable; under this assumption, the scheme is semantically secure. (Efficient decryption is possible since it is easily computable whether a ciphertext is a square modulo p and modulo q- the encrypted message is 0 iff the ciphertext is a square modulo both of these.)

We can note that the QR scheme is strongly HE w.r.t the XOR operation. Here, note that $\mathsf{E}(0)$ and $\mathsf{E}(1)$ are distributions, and so the equality statements indicate that the resulting distributions are equivalent.

- $\mathsf{E}(0) \cdot \mathsf{E}(0) \mod N = \mathsf{E}(0)$, since for any $x, y \in \mathbb{Z}_N^*$, $(x^2 \mod N) \cdot (y^2 \mod N) \equiv (xy)^2 \mod N$ for a $xy \in \mathbb{Z}_N^*$
- $\mathsf{E}(1) \cdot \mathsf{E}(1) \mod N = \mathsf{E}(0)$, since for any $x, y \in \mathbb{Z}_N^*$, $(-x^2 \mod N) \cdot (-y^2 \mod N) \equiv (xy)^2 \mod N$ for a $xy \in \mathbb{Z}_N^*$
- $\mathsf{E}(0) \cdot \mathsf{E}(1) \mod N = \mathsf{E}(1)$, since for any $x, y \in \mathbb{Z}_N^*$, $(x^2 \mod N) \cdot (-y^2 \mod N) \equiv -(xy)^2 \mod N$ for a $xy \in \mathbb{Z}_N^*$

4 Relationship Between Public Key and Private Key Cryptography

Very recent work from Ron Rothblum shows how to transform any weakly homomorphic private-key encryption scheme with multiple message security into a public-key encryption scheme.

4.1 Warm Up Result: Strong Homomorphic Private Key w.r.t Identity Function Implies Public Key

As a warm-up to this result, we instead show that a private-key encryption scheme that is strongly homomorphic with respect to the identity function, and that has multiple message security, can be used to construct a public-key encryption scheme. We begin with the homomorphic private-key system defined by the tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{E}, \mathsf{D}, \mathsf{Eval})$. Since this is a private-key scheme, Eval is no longer given a key, and only takes a description of a circuit along with a collection of ciphertexts.

Let $\mathcal{I}$ represent a circuit describing the identity function. The strong homomorphic property over the identity function ensures for any message M and ciphertext $C \xleftarrow{\$} \mathsf{E}(sk, M)$, the distributions $(\mathsf{Eval}(\mathcal{I}, C), C)$ and $(\mathsf{E}(sk, M), C)$ are equal over the probability space of ciphertexts returned by $\mathsf{E}(sk, M)$ and $\mathsf{Eval}(\mathcal{I}, C)$; in other words, $\mathsf{Eval}(\mathcal{I}, C)$ should look like a freshly generated ciphertext of M.

We can now generate the new public-key encryption algorithm $(\mathsf{Gen}', \mathsf{E}', \mathsf{D}')$ as follows.

- $\mathsf{Gen}'(1^k)$ returns $sk \xleftarrow{\$} \mathsf{Gen}(1^k)$, and $pk \leftarrow (C_0, C_1)$ where $C_0 \xleftarrow{\$} \mathsf{E}(sk, 0)$ and $C_1 \xleftarrow{\$} \mathsf{E}(sk, 1)$.
- $\mathsf{E}'(pk, b)$ computes $C \xleftarrow{\$} \mathsf{Eval}(\mathcal{I}, \mathsf{E}(sk, b))$.
- $\mathsf{D}' = \mathsf{D}$.

Note that the ciphertexts generated by E' take advantage of the homomorphism with respect to the identity function to generate ciphertexts that are distributed identically to freshly generated ciphertexts in the original private key system, and so inherit their security properties!

4.2 Homomorphic Private Key Encryption Scheme w.r.t. XOR Implies Public Key Encryption Scheme

Recall that Rothblum's result doesn't start with a *strong* homomorphic scheme, but rather a weak one; as weak homomorphism with respect to XOR (the basis for Rothblum's theorem) is not as powerful as strong homomorphism with respect to the identity function, we need new methods to create the public-key encryption scheme.

Theorem 5 *Any multiple-message semantically secure private-key encryption scheme that is weakly homomorphic with respect to XOR can be transformed into a semantically secure public-key encryption scheme.*

To prove his theorem, we use any private-key scheme $(\mathsf{Gen}, \mathsf{E}, \mathsf{D}, \mathsf{Eval})$ satisfying the Theorem hypothesis to construct a public-key scheme $(\mathsf{Gen}', \mathsf{E}', \mathsf{D}')$ as follows:

- $\mathsf{Gen}'(1^k)$ returns the sk generated by $\mathsf{Gen}(1^k)$, but generates pk in a different fashion: let $\ell = 3m + k$ where $m = B(k)$ upper-bounds the size of Eval's output; choose $r \xleftarrow{\$} \{0,1\}^\ell$, and let $pk = (r, C_i \in \mathsf{E}(sk, r_1), \ldots, C_r \in \mathsf{E}(sk, r_\ell))$. For notational convenience, let pk_i denote the i^{th} ciphertext C_i in pk.
- $\mathsf{E}'(pk, b)$ selects $s \xleftarrow{\$} \{y \in \{0,1\}^\ell | \langle y, r\rangle \equiv b \mod 2\}$, then returns $C \xleftarrow{\$} \mathsf{Eval}\,(\mathrm{XOR}, pk_i \forall i : s_i = 1)$.
- $\mathsf{D}'(sk, C) = \mathsf{D}(sk, C)$.

To observe the correctness property for this scheme, note that $\mathrm{XOR}_{s_i=1}(r_i) = \langle s, r\rangle \mod 2 = b$, and so this homomorphic operation gives a ciphertext decrypting to b in the original private-key system.

Before giving a proof of the security properties of the scheme, we give a high level description of a reduction to the security of the underlying private-key scheme. To do this, we consider improperly formed public keys that still contain random r, but instead of encryptions of the bits of r now contain encryptions of all 0's. With these improperly formed public keys, the ciphertext no longer depends on r, and contains a limited amount of information about s. Intuitively, which will be proved using the leftover hash lemma, it can be shown that in this case no adversary can have more than a negligible advantage in guessing whether the underlying message is 0 or 1. By the multiple message security of the original scheme, these new public keys should be computationally indistinguishable from properly formed public-keys, and so it must be the case that also with properly formed public-keys an adversary can not distinguish whether the underlying message is 0 or 1.

A more formal proof of the main Theorem 6 proceeds through two Lemmas. Let the private-key system be defined by the tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{E}, \mathsf{D}, \mathsf{Eval})$. Recall that the properties of weak homomorphism guarantee that there exists a polynomial $B(k)$ that upper-bounds the size of ciphertext produced by Eval- here, we use $m = B(k)$ to represent this upper-bound. We let $\ell = 3m + k$ be the number of ciphertexts in the public key. By setting $\ell > m$, when we combine order ℓ ciphertexts using the XOR homomorphism, information must be compressed to fit into the resulting special ciphertext.

We let Rothblum's constructed public-key system described above be defined by algorithms $(\mathsf{Gen}', \mathsf{E}', \mathsf{D}')$.

Lemma 6 *Assume a private-key encryption scheme* $(\mathsf{Gen}, \mathsf{E}, \mathsf{D}, \mathsf{Eval})$ *satisfying the Hypothesis of Theorem 6, and let* $(\mathsf{Gen}', \mathsf{E}', \mathsf{D}')$ *be defined as above.*

Let a public key pk be incorrectly formed by generating $r \xleftarrow{\$} \{0,1\}^\ell$, and setting $pk = (r, c_1 \leftarrow \mathsf{E}(sk, 0), \ldots, c_l \leftarrow \mathsf{E}(sk, 0))$. Then for any adversary $\mathcal{A} \in \mathsf{PPT}$, $\forall c > 0$ $\exists K$ such that $\forall k > K$, when $m \leq B(k)$ and $\ell = 3m + k$,

$$\Pr[s, r \xleftarrow{\$} \{0,1\}^\ell; sk \leftarrow \mathsf{Gen}(1^k); pk \leftarrow (r, c_1 \leftarrow \mathsf{E}(sk, 0), \ldots, c_l \leftarrow \mathsf{E}(sk, 0)) :$$
$$\mathcal{A}\,(pk, \mathsf{Eval}(XOR, pk_i \forall i : s_i = 1)) = \langle s, r \rangle] < \tfrac{1}{2} + 3 \cdot 2^{-\frac{\ell}{2}+m}.$$

Proof:

Note that with the incorrectly formed pk, $\mathsf{Eval}(\mathrm{XOR}, pk_i \forall i : s_i = 1)$ does not depend on the randomly chosen r at all- once given pk, the adversary $\mathcal{A}$ must guess $\langle s, r \rangle$ using r, and m bits of information about s, where m is the maximum size ciphertext output by Eval.

To proceed, we use the Leftover Hash Lemma corollary 4. We use $f(s)$ to denote $\mathsf{Eval}(pk, s)$ for a fixed pk, with $f : \{0,1\}^\ell \to \{0,1\}^m$. Since $h_r(s) = \langle s, r \rangle$ is a pair-wise independent hash function family, for every $\alpha \in \{0,1\}^m$, for $f : \{0,1\}^\ell \to \{0,1\}^m$ the distribution $(r, h_r(s))$ conditioned on $f(s) = \alpha$ and the distribution (r, b) for $b \xleftarrow{\$} \{0,1\}$ are $2\sqrt{\frac{2}{|f^{-1}(\alpha)|}}$-close in statistical distance. We will also use the fact that

$$\sum_{\alpha \in \{0,1\}^m} |f^{-1}(\alpha)| = 2^\ell,$$

ie the sum of the size of pre-images of f over the range of f is the size of the domain of f.

Then

$$\Pr[r, s \xleftarrow{\$} \{0,1\}^\ell) : \mathcal{A}(r, f(s)) = \langle s, r \rangle]$$
$$= \sum_{\alpha \in \{0,1\}^m} \Pr[s \xleftarrow{\$} \{0,1\}^\ell : f(s) = \alpha] \cdot \Pr[r, s \xleftarrow{\$} \{0,1\}^\ell : A(r, \alpha) = \langle s, r \rangle \,|f(s) = \alpha]$$
$$\leq \sum_{\alpha \in \{0,1\}^m} \frac{|f^{-1}(\alpha)|}{2^\ell} \cdot \left(\Pr[r \xleftarrow{\$} \{0,1\}^\ell; b \xleftarrow{\$} \{0,1\} : \mathcal{A}(r, \alpha) = b] + 2\sqrt{\frac{2}{|f^{-1}(\alpha)|}} \right)$$
$$\leq \tfrac{1}{2} \sum_{\alpha \in \{0,1\}^m} \frac{|f^{-1}(\alpha)|}{2^\ell} + \sum_{\alpha \in \{0,1\}^m} \frac{2\sqrt{2|f^{-1}(\alpha)|}}{2^\ell}$$
$$\leq \tfrac{1}{2} + 3 \cdot 2^{-\frac{\ell}{2}+m}.$$

□

Lemma 7 *No adversary $\mathcal{B} \in \mathsf{PPT}$ can distinguish encryptions made using E' with $s \xleftarrow{\$} \{0,1\}^\ell$ of a bit $b = \langle s, r \rangle$ with non-negligible advantage.*

Proof:

Assume for the contradiction that there $\exists \mathcal{B} \in \mathsf{PPT}$ that *can* - then there exists a polynomial $p(\cdot)$ and infinitely many k such that

$$\Pr[sk \leftarrow \mathsf{Gen}(1^k); r \xleftarrow{\$} \{0,1\}^\ell; pk = (r, \mathsf{E}(sk, r_1), \ldots, \mathsf{E}(sk, r_\ell)); b \xleftarrow{\$} \langle s, r\rangle \{0,1\} : \mathcal{B}(pk, \mathsf{E}'(pk, b)) = b] > \frac{1}{2} + \frac{1}{p(k)}$$

By Lemma 7, if $pk = (r, \mathsf{E}(sk, r_1), \ldots, \mathsf{E}(sk, r_\ell))$,

$$\Pr[sk \leftarrow \mathsf{Gen}(1^k); r \xleftarrow{\$} \{0,1\}^\ell; pk = (r, \mathsf{E}(sk, 0), \ldots, \mathsf{E}(sk, 0)); b \xleftarrow{\$} \langle s, r\rangle \{0,1\} : \mathcal{B}(pk, \mathsf{E}'(pk, b)) = b] < \frac{1}{2} + 3 \cdot 2^{-\frac{\ell}{2}+m}$$

Then given $C^* \leftarrow \mathsf{Eval}(\mathrm{XOR}, pk_i \forall i : s_i = 1)$, $\mathcal{B}$ can be used to distinguish between encryptions of $r \xleftarrow{\$} \{0,1\}^\ell$ and of $r \leftarrow 0^\ell$. In particular, have the distinguisher receive Y which is comprised of r and either an encryption of the ℓ bits of r, or encryptions of ℓ copies of 0. Have the distinguisher choose $s \xleftarrow{\$} \{0,1\}^\ell$, compute $\mathcal{B}(Y, \mathsf{Eval}(\mathrm{XOR}, pk_i \forall i : s_i = 1))$, and then output 1 if $\mathcal{B}$ returns $\langle s, r\rangle$, and 0 otherwise. This distinguisher has non-negligible advantage for our choice of $\ell = 3m + k$, contradicting the multiple message security of the original private key scheme. □

Since no adversary's can distinguish a random bit with non-negligible advantage, we get semantic security, as desired.

Remark: Yevgeniy Dodis noted that actually one only needs a weakly homomorphic private-key encryption scheme with respect to an extractor - Rothblum's scheme instantiates this with the dot product. (Extractors to be defined later in the course.)