DEGREE PROJECT IN TECHNOLOGY,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2023

Homomorphic Encryption

KTH Thesis Report

Daniel Sanaee

KTH ROYAL INSTITUTE OF TECHNOLOGY
MATHEMATICS

Authors

Daniel Sanaee <dsanaee@kth.se>
Mathematics
KTH Royal Institute of Technology

Place for Project

Stockholm, Sweden

Examiner and supervisor

Johan Håstad
KTH Royal Institute of Technology

Abstract

Write an abstract. Introduce the subject area for the project and describe the problems that are solved and described in the thesis. Present how the problems have been solved, methods used and present results for the project. Use probably one sentence for each chapter in the final report.

The presentation of the results should be the main part of the abstract. Use about ½ A4-page. English abstract

Keywords

Template, Thesis, Keywords ...

Abstract

Svenskt abstract

Nyckelord

Mall, Examensarbete, Nyckelord ...

Acknowledgements

Write a short acknowledgements. Don’t forget to give some credit to the examiner and supervisor.

Acronyms

Contents

Introduction

Computation theory

In this section, it is assumed that the reader has prior knowledge of deterministic- and probabilistic Turing machines as a model of computation (an excellent introduction can be found in [Gol01]). We introduce an alternative model of computation based on sets of circuits for the purpose of protection against stronger adversaries. We include the basics needed to understand cryptography and homomorphic encryption in particular.

Digital logic

Definition 1 (Circuit)**.** For $n, m \in \mathbb{N}$ and any field $(\mathbb{F}, +, \times)$, an arithmetic circuit is a vector-valued polynomial function $C \colon \mathbb{F}^n \to \mathbb{F}^m$.

A circuit C is represented by a finite directed acyclic graph with n source nodes (the n inputs) and m sink nodes (the m outputs). The internal nodes of the circuit are called gates. For more details about the structure of a circuit, see [Gol08] or [MF21]. The number of nodes in C is called its *size* and is denoted $|C|$. The longest path in C is called its *depth*. A circuit is called *Boolean* when the field is $\mathbb{F}_2$ and each gate takes at most 2 inputs. Boolean circuits and arithmetic circuits are equivalent in the sense that the set of functions that can be computed by an arithmetic circuit is equal to the set of functions that can be computed by a Boolean circuit.[1] If the input is a string of bits, we assume the circuit is Boolean and if the input is a tuple of arbitrary inputs (messages) in the field, then we assume an arithmetic circuit.

We consider circuits as algorithms and use them as an alternative approach to the

[1] A Boolean circuit is an arithmetic circuit. Conversely, any Boolean circuit can be simulated by converting every gate to XOR and AND gates and using $\mathrm{XOR}(a, b) = a + b - 2 * a * b$, $\mathrm{AND}(a, b) = a * b$.

traditional Turing machine model of computation.[2] Notice that any given circuit, C, can only compute on inputs of the same length whereas a Turing machine M takes inputs of any size n. However, a circuit always halts on a given input whereas a Turing machine may not. For the purpose of our discussion relating to cryptography, we assume every Turing machines halts unless otherwise stated. To allow circuits to handle arbitrary length inputs we consider families of circuits.

Definition 2 (Circuit family [MF21])**.** A circuit family $C = \{C_n\}_{n\in\mathbb{N}}$ is an indexed set of circuits $C_n \colon \mathbb{F}^{n+r} \to \mathbb{F}^m$ where $r, m = \text{poly}(n)$.

For any input x with length n, $C(x) \overset{\text{def}}{=} C_n(x)$. For each circuit $C_n \in C$, r represents the random coins used. If $r = 0$ for all n then C is a deterministic circuit family. A circuit family is said to have polynomial-size if there exists a polynomial p such that $|C_n| < p(n)$ for all n.

Complexity classes

Definition 3 (Complexity Class P)**.** P is the set of languages $\mathscr{L}$ such that there exists a deterministic polynomial-time Turing machine M satisfying $M(x) = 1 \iff x \in \mathscr{L}$

Definition 4 (Complexity Class BPP)**.** BPP is the set of languages $\mathscr{L}$ such that there exists a probabilistic polynomial-time (PPT) Turing machine M satisfying

$$\Pr[M(x) = 1] \geq 2/3 \text{ if } x \in L$$
$$\Pr[M(x) = 1] \leq 1/3 \text{ if } x \notin L$$

Definition 5 (Complexity Class P/poly)**.** P/poly is the set of languages $\mathscr{L}$ such that there exists a polynomial-size circuit family C satisfying $C(x) = 1 \iff x \in \mathscr{L}$

Informally speaking, circuit families is a stronger model of computation than the PPT Turing machine model in the sense that if there exists a PPT Turing machine for deciding a problem, then there also exists a circuit family that can decide the same problem. The formal statement is as follows:

[2] The reason for this alternative model is to assume adversaries are computationally ”stronger”. See Theorem 1.

Theorem 1. $\mathrm{P} \subseteq \mathrm{BPP} \subsetneq \mathrm{P/poly}$

The first inclusion follows from the fact that if there exists a deterministic Turing machine that decides a language, then that same machine can be seen as a PPT machine that ignores a given input sequence of coin tosses. For the second inclusion, consider a language $\mathscr{L} \in \mathrm{BPP}$ and a corresponding PPT machine M for $\mathscr{L}$. Then, for any given input x_n with length n, at least $2/3$ of the set of all possible coin toss sequences are good (good r means $M(x_n; r) = 1 \iff x_n \in \mathscr{L}$). This means that there exists at least 1 sequence of coin tosses that yields the correct result for $2/3$ of the possible inputs of length n. By using an alternative (but equivalent) definition of BPP, it can be shown that there actually exist a coin toss sequence, r_n, that yields the correct result for all inputs of length n (see [Adl78; Gol01] for more detail). Consider circuit $C_n \colon \{0,1\}^{n+|r_n|} \to \{0,1\}$ with r_n hardcoded as inputs where C_n simulates M using r_n, i.e., $C_n(x_n) = M(x;r)$. Therefore $C_n(x) = 1 \iff x \in \mathscr{L}$ and C decides $\mathscr{L}$.

Interestingly the first inclusion is speculated to be set equivalence [AB09, pp. 126], meaning that a deterministic machine could decide the same languages as a probabilistic one. The second inclusion is proper since every unary language is in P/poly whereas undecidable ones are not in BPP (see [AB09, pp. 110] for details). In this sense, circuit families are a stronger model of computation than PPT Turing machines. We capture this notion with uniformity.

Definition 6 (Uniform circuit family)**.** A circuit family $\{C_n\}_{n\in\mathbb{N}}$ is uniform if there exists a polynomial-time Turing machine M such that $M(1^n)$ outputs the description of C_n for all $n \in \mathbb{N}$.

A uniform circuit family is polynomial size. The converse is not necessarily true. A family that is not uniform is said to be a non-uniform circuit family. Note that Turing machines are at least as strong as uniform circuit families. More formally, if a uniform circuit family decides $\mathscr{L}$ then there exists a polynomial-time Turing machine that decides $\mathscr{L}$.[3] Simply construct the polynomial-time Turing machine that given any input $x \in \mathscr{L}$, first generates a description of $C_{|x|}$ and then simulates $C_{|x|}(x)$. In other words, the non-uniform circuit families are stronger than the polynomial-time Turing machines.

[3] The converse is also true, meaning deterministic polynomial-time Turing machines are exactly as powerful as uniform circuit families. See [AB09, pp. 111] for details

Probability Theory

Homomorphic encryption schemes are based on noise. As we will see in Subsection 5.1, noise can be sampled from discrete spaces in accordance with a distribution. Since distributions are central in cryptography, it is important they are understood. A distribution is a probability measure on a measurable space $(S, \mathcal{S})$. Typically, probability distributions are associated with random variables; however, in the absence of random variables, distributions are understood as a specified measure function on the given measurable space (See [Kal21, pp. 83]).

Definition 7 (Discrete distribution measure of a random variable)**.** Consider a probability space $(\Omega, \mathcal{F}, \Pr)$ and a discrete measurable space $(S, \mathcal{S})$. Let $X \colon \Omega \mapsto S$ be a random variable. The discrete distribution measure of X, or simply *distribution* of X, χ is defined as follows

$$\begin{aligned} \chi \colon \mathcal{S} &\to [0,1] \\ \{x\} &\mapsto \Pr[X = x] \end{aligned}$$

We say that χ is the distribution or *law* of X, denoted $\mathcal{L}(X)$.

Remark. Since χ is a measure on a discrete space, the description of the singletons are sufficient. More explicitly, $\chi(A) = \sum_{x \in A} \Pr[X = x]$ for $A \in \mathcal{S}$

A distribution measure for a random variable is a probability measure on the sample space $(S, \mathcal{S})$, as opposed to on the outcome space $(\Omega, \mathcal{F})$. For a given probability space, any random variable X defined on it has a distribution associated with it. We write $x \leftarrow X$ or $x \leftarrow \chi$ for sampling the outcome x from X assuming χ is its distribution. When X is a space we mean that x is uniformly sampled from X.

Definition 8 (Ensembles)**.** Let I be a countable index set. A *probability ensemble* indexed by I (or just ensemble) is a sequence of random variables $(X_i)_{i \in I}$. A *distribution* *ensemble* indexed by I is a sequence of distributions $(\chi_i)_{i\in I}$

In this paper we will exclusively use $\mathbb{N}$ as the index set. For example, the encryption function is a random variable (PPT algorithm), meaning that a single message m corresponds to many valid ciphertexts. By varying the security parameter of the scheme we construct the probability ensemble $\{Enc(pk, m)\}_{pk\in\mathbb{N}}$

Definition 9 (Statistical distance)**.** Let S_n be finite set for all $n \in \mathbb{N}$ and let $X = (X_n\colon \Omega_n \to S_n)_{n\in\mathbb{N}}$, $Y = (Y_n\colon \Omega_n \to S_n)_{n\in\mathbb{N}}$ be two ensembles. The *statistical distance* is defined as

$$\Delta_{X,Y}(n) \overset{\text{def}}{=} \frac{1}{2}\sum_{\alpha\in S_n} |\Pr[X_n = \alpha] - \Pr[Y_n = \alpha]|.$$

Let $\chi = \{\chi_n\}_{n\in\mathbb{N}}$, $\psi = \{\psi_n\}_{n\in\mathbb{N}}$ be two discrete distribution ensembles on the same measurable spaces $\{(S_n, \mathcal{S}_n)\}_{n\in\mathbb{N}}$. The statistical distance is defined as

$$\Delta_{\chi,\psi}(n) \overset{\text{def}}{=} \frac{1}{2}\sum_{\alpha\in S_n} |\chi_n(\{\alpha\}) - \psi_n(\{\alpha\})|$$

Remark. This definition for distribution ensembles assumes every singleton is measurable. A more general definition is $\Delta(\chi, \psi) \overset{\text{def}}{=} \sup_{A\in\mathcal{S}} |\chi(A) - \psi(A)|$

Definition 10 (Negligible function)**.** Let $f\colon \mathbb{N} \to [0, 1]$. f is negligible if for all positive polynomials $p(\cdot)$, there exists an N such that for all $n > N$, $f(n) < \frac{1}{p(n)}$. We say $f = \text{negl}(n)$.

Definition 11 (Perfectly indistinguishable)**.** Two ensembles (probability or distribution) X and Y are *perfectly indistinguishable* if $\Delta_{X,Y}(n) = 0$ for all n.

Definition 12 (Statistically indistinguishable)**.** Two ensembles (probability or distribution) X and Y are *statistically indistinguishable* if $\Delta_{X,Y}(n) = \text{negl}(n)$.

A desirable property of encryption schemes is to make it unfeasible for adversaries to distinguish encryptions. For two random variables (or distributions) X and Y, we want every adversary C to not be able to distinguish samples from X and Y. More precisely, an adversary tasked with identifying whether given samples are from X (C outputs 0) or from Y (C outputs 1) should have roughly the same probability of success irrespective of which the samples are generated from. The formal definition is as follows:

Definition 13 (Computationally indistinguishable [Gol01])**.** Two ensembles (or distribution ensembles) X and Y are computationally indistinguishable if, for every polynomial-size circuit family $C = \{C_n\}_{n\in\mathbb{N}}$,

$$\mathrm{Adv}_{X,Y}(n) \overset{\text{def}}{=} |\Pr[C(X_n) = 1] - \Pr[C(Y_n) = 1]| = \mathrm{negl}(n).$$

$C(X_n), C(Y_n)$ means that the adversary has access to random oracle returning a sample from X_n and Y_n respectively. The point is that, for every adversary C guessing the samples are from one of the random variables or distributions (1 in this case represents from Y_n), the probability of being correct is, up to negligible error, equal to the probability of being incorrect. For example, an adversary always guessing 1 has 100% probability of being incorrect when given samples from X_n and 100% probability of being correct when given samples from Y_n. The intuition is that the advantage in distinguishing the ensembles goes to zero fast (faster than the inverse of all polynomials).

Perfect distinguishability implies statistical indistinguishability as 0 is negligible. Statistical indistinguishability implies computational indistinguishability. To see why, assume $\Delta_{X,Y}(n) = \mathrm{negl}(n)$ and consider any circuit family C. Since X_n and Y_n are defined on the same sample space we have

$$\begin{aligned}
\mathrm{Adv}_{X,Y}(n) &= |\Pr(X_n) = 1] - \Pr[C(Y_n) = 1]| \\
&\leq \sum_{\alpha\in S_n} |\Pr[C(X_n = \alpha) = 1 \mid X_n = \alpha] \Pr[X_n = \alpha] \\
&\qquad\quad - \Pr[C(Y_n = \alpha) = 1 \mid Y_n = \alpha] \Pr[Y_n = \alpha]| \\
&\leq \sum_{\alpha\in S_n} |\Pr[X_n = \alpha] - \Pr[Y_n = \alpha]| = 2\Delta_{X,Y}(n) = \mathrm{negl}(n)
\end{aligned}$$

where the first inequality is due to the triangle inequality and the second is due to conditional probability being upper bounded by 1. Intuitively, there does not exist enough distinguishable outcomes to distinguish the ensembles even if there exists an adversary that always correctly recognize an outcome from X and Y respectively.

Distributions for noise in HE

To add discrete noise to ciphertexts, it is typical to use the discretized Gaussian distribution introduced by Micciancio and Regev [MR07]. Since this distribution is used so frequently, we give a brief presentation of it here. A continous Gaussian distribution in

$\mathbb{R}^n$ centered at $\vec{c}$ where each coordinate is independently sampled from $\mathcal{N}(0, \sigma^2)$, then the multivariate Gaussian distribution is simplified to $\rho_{\vec{c}}(\vec{x}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\|\vec{x} - \vec{c}\|^2\right)$. By introducing the scale factor $s \overset{\text{def}}{=} \sqrt{2\pi}\sigma$ we get $\rho_{s,\vec{c}}(\vec{x}) = \frac{1}{s^n} \exp\left(-\frac{\pi}{s^2}\|\vec{x} - \vec{c}\|^2\right)$. The goal is to discretize the gaussian over lattices, see Section 5, by restricting the points to lattice points. For a given lattice $\Lambda \subset \mathbb{R}^n$ we define the normalization constant $\rho_{s,\vec{c}}(\Lambda) \overset{\text{def}}{=} \sum_{\vec{x} \in \Lambda} \rho_{s,\vec{c}}(\vec{x})$.

Definition 14 (Discrete Gaussian distribution)**.** Let $\Lambda \subset \mathbb{R}^n$ be a lattice. Then the discrete gaussian distribution is defined as

$$\forall \vec{x} \in \Lambda\colon \quad \mathrm{D}_{\Lambda,s,\vec{c}}(\vec{x}) \overset{\text{def}}{=} \frac{\rho_{s,\vec{c}}(\vec{x})}{\rho_{s,\vec{c}}(\Lambda)}.$$

Definition 15 (β-bounded distribution)**.** Let $\chi = \{\chi_n\}_{n \in \mathbb{N}}$ be a distribution ensemble over the integers. χ is β-bounded if $x \leftarrow \chi_n \implies \Pr[|x| > \beta] = \mathrm{negl}(n)$

In this paper, every distribution is assumed to be β-bounded for some $\beta = q(n)\alpha(n)$.

Cryptographic primitives

In the following definitions we let the message space be denoted $\mathcal{X}$, the ciphertext space be denoted $\mathcal{Y}$, and the key space be denoted $\mathcal{K} = \mathcal{K}_{pk} \times \mathcal{K}_{sk}$.

Definition 16 (Encryption scheme)**.** A correct asymmetric encryption scheme $\mathcal{E} = (\text{KeyGen}, \text{Enc}, \text{Dec})$ is a triple of algorithms satisfying the following:

- $\text{KeyGen} : \{1\}^* \to \mathcal{K}$ is PPT given by $1^\lambda \mapsto (pk, sk)$.
- $\text{Enc} : \mathcal{K}_{pk} \times \mathcal{X} \to \mathcal{Y}$ is PPT given by $(pk, m) \mapsto c$.
- $\text{Dec} : \mathcal{K}_{sk} \times \mathcal{Y} \to \mathcal{X}$ is deterministic and satisfies $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda) \implies \text{Dec}(sk, \text{Enc}(pk, m)) = m$.

Remark. This paper is only concerned with encryption schemes where decryption always works (correct) and where more than one key is used (asymmetric). Thus, every encryption scheme is assumed to be a correct asymmetric encryption scheme. Furthermore, the decryption algorithm can be considered a PPT algorithm that with probability 1 outputs the correct message for the correct secret key. In other words, the algorithm ignores the input coin toss sequence.

In this paper, we consider homomorphic encryption (HE) schemes. These schemes include a fourth algorithm, Eval, called the evaluation algorithm which is used by the server calculating on encrypted data.

Definition 17 ($\mathcal{C}$-homomorphic encryption scheme)**.** An encryption scheme $\mathcal{E}$ is a $\mathcal{C}$-homomorphic encryption scheme for the set of circuits $\mathcal{C}$ if there exists an extra algorithm Eval such that for any $C \in \mathcal{C}$ taking t inputs the following condition holds:

- $\text{Eval} \colon \mathcal{K}_{pk} \times \mathcal{C} \times \mathcal{Y}^* \to \mathcal{Y}$ is PPT and satisfies $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda) \implies \text{Dec}(sk, \text{Eval}(pk, C, \langle \text{Enc}(pk, m_1), \ldots, \text{Enc}(pk, m_t) \rangle)) = C(m_1, \ldots, m_t)$

$$\begin{tikzcd}
\mathcal{Y}^t \arrow[r, "\mathrm{Eval}(pk{,}C{,}\langle c_1{,}...{,}c_t\rangle) = c_f"] \arrow[d, "\mathrm{Dec}(sk{,}\langle c_1{,}...{,}c_t\rangle)"'] & \mathcal{Y} \arrow[d, "\mathrm{Dec}(sk{,}c_f)"] \\
\mathcal{X}^t \arrow[r, "C(m_1{,}...{,}m_t)"'] & \mathcal{X}
\end{tikzcd}$$

Figure 1: The decryption homomorphism. The path through $\mathcal{Y}$ represents computing before decrypting. The path through $\mathcal{X}^t$ represents decrypting before computing.

We say $\mathcal{E}$ can evaluate all circuits in $\mathcal{C}$ and is $\mathcal{C}$-homomorphic.

The evaluation algorithm runs the ciphertexts through the permissible circuit while also satisfying the requirement that decrypting the resulting ciphertext yields the same result as the plaintexts running through the circuit. To its disposal, the evaluation algorithm is given the public key. The ciphertexts returned by the Eval algorithm are called *evaluated ciphertexts* (suggesting the circuit has evaluated the ciphertexts) and those returned by the encryption algorithm are called *fresh ciphertexts*. Remark that correctness is only guaranteed if the Eval algorithm is given fresh ciphertexts. If the circuit corresponds to the computable function f acting on a vector c of ciphertexts, we denote the evaluated ciphertexts c_f (i.e., $c_f \overset{\text{def}}{=} \mathrm{Eval}(pk, f, c)$). Similarly, for the vector m of plaintexts, we denote the evaluated plaintexts m_f (i.e., $m_f \overset{\text{def}}{=} f(m)$). Thus, the condition for the Eval algorithm can be written $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda) \implies \mathrm{Dec}(sk, c_f) = m_f$. In a homomorphic encryption scheme that supports one addition and multiplication of fresh ciphertexts, the decryption function is a ring homomorphism. Consider a valid key pair (sk, pk) which are, for notational simplicity, hard-wired into the decryption and evaluate functions respectively, plaintext ciphertext pairs $(m_1, c_1 = \mathrm{Enc}_{pk}(m_1))$, $(m_2, c_2 = \mathrm{Enc}_{pk}(m_2))$ and circuits $C_+(m_1, m_2) \overset{\text{def}}{=} m_1 + m_2$ and $C_\times(m_1, m_2) \overset{\text{def}}{=} m_1 \times m_2$ that can be evaluated by the scheme.

$$\begin{aligned}
\mathrm{Dec}_{sk}(c_1 + c_2) &= \mathrm{Dec}_{sk}(\mathrm{Eval}_{pk}(C_+, \langle c_1, c_2\rangle)) = C_+(m_1, m_2) = m_1 + m_2 \\
&= \mathrm{Dec}_{sk}(c_1) + \mathrm{Dec}_{sk}(c_2) \\
\mathrm{Dec}_{sk}(c_1 \times c_2) &= \mathrm{Dec}_{sk}(\mathrm{Eval}_{pk}(C_\times, \langle c_1, c_2\rangle)) = C_\times(m_1, m_2) = m_1 \times m_2 \\
&= \mathrm{Dec}_{sk}(c_1) \times \mathrm{Dec}_{sk}(c_2)
\end{aligned}$$

The main idea behind a homomorphic encryption scheme is to give a server encrypted data so that it can compute on that data and return the answer in encrypted form.

However, the definition provided allows for trivial homomorphic encryption schemes where the server does nothing. More specifically, consider any set of circuits $\mathcal{C}$ and let $\text{Eval}(pk, C, \langle c_1, \ldots, c_t \rangle) = (C, \langle c_1, \ldots, c_t \rangle)$. Eval takes a description of a circuit and a tuple of ciphertexts, one for each input wire of the circuit, and simply outputs the description of the circuit together with the given tuple. Clearly, Eval runs in polynomial time. Consider a decryption algorithm that, if given an input of this form, first decrypts the t ciphertexts and then computes m_f using C. To ensure that the server actually processes the given inputs we introduce compactness.

Definition 18 (Compactness)**.** A $\mathcal{C}$-homomorphic encryption scheme is compact if there exists a polynomial $p(\lambda)$ such that for all $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$, for every $C \in \mathcal{C}$ taking any number t inputs and any $c \in \mathcal{Y}^t$, the size of the output $\text{Eval}(pk, C, \langle c_1, \ldots, c_t \rangle)$ is less than $p(\lambda)$. We say that the scheme compactly evaluates $\mathcal{C}$.

For a compact $\mathcal{C}$-homomorphic encryption scheme, the size of the output is independent of the circuit function used. In particular, the previous, trivial homomorphic encryption scheme where $\text{Eval}(pk, C, \langle c_1, \ldots, c_t \rangle) = (C, \langle c_1, \ldots, c_t \rangle)$ is not compact for any set of circuits with unbounded circuit size, which includes circuit families with circuits that do not ignore all except for constantly many inputs, meaning essentially every application of a HE scheme.

Definition 19 (Fully Homomorphic Encryption (pure FHE))**.** Let $\mathcal{C}$ be the class of all circuits. An encryption scheme $\mathcal{E}$ is a fully homomorphic encryption (pure FHE) scheme if it is $\mathcal{C}$-homomorphic and compactly evaluates $\mathcal{C}$.

For a scheme to be fully homomorphic it is required that it can evaluate circuits of arbitrary size. Many times it suffices to consider only circuits of a beforehand specified depth, L, as any deeper circuits are irrelevant to the application. The following definition capture schemes that can evaluate any set of circuits with depths bounded by the client.

Definition 20 (Leveled fully homomorphic encryption (leveled FHE))**.** An encryption scheme $\mathcal{E}$ with the KeyGen algorithm modified is a leveled fully homomorphic encryption scheme if it satisfies the following:

- $\text{KeyGen} : \{1\}^* \times \{1\}^* \to \mathcal{K}$ is PPT given by $(1^\lambda, 1^L) \mapsto (pk, sk)$.
- Let $\mathcal{C}_L$ be the set of circuits with depth less than or equal to L. Then $\mathcal{E}$ is $\mathcal{C}_L$-homomorphic.

- $\mathcal{E}$ compactly evaluates the set of all circuits.

Remark. Notice that the length of the evaluated ciphertexts in a leveled FHE scheme is independent of the depth. Also note that for any circuit C, there exists an L such that a leveled FHE scheme can evaluate it.

A potential point of contention is the purpose of a pure FHE scheme in the presence of a leveled FHE scheme; why do we not simply choose an L such that the leveled FHE scheme can evaluate any given circuit C?

Possible answer? ask Johan
It's not always clear what the depth of the circuit is. Therefore, it may be needed to use an excessively large depth parameter L. In a leveled scheme, an increasing L allows for longer keys and ciphertext lengths and consequently a performance overhead. In contrast, a pure FHE scheme is independent of the depth, meaning ciphertexts lengths are only bounded by the security parameter.

Security definitions

In this paper, semantic security refers to security against chosen-plaintext attack (CPA). The definition relates to the following game where the challenger possess the secret key and the player is the adversary trying to break the scheme. Consider encryption scheme (KeyGen, Enc, Dec, Eval) and polynomial-size Boolean circuit family $C = \{C_n\}_{n \in \mathbb{N}}$. The CPA game is defined with the Boolean function $\text{CPA}_C(\lambda)$ as follows:

1. **Setup**: Challenger samples $pk \leftarrow \text{KeyGen}(1^\lambda)$ and sends it to player.
2. **Choose**: Player C selects two distinct plaintext messages $(m_0, m_1) \leftarrow C(pk)$ of the same length, and sends them to the challenger.
3. **Encrypt**: The challenger randomly picks a bit $b \in \{0, 1\}$ and encrypts the message m_b. The encrypted message $c \overset{\text{def}}{=} \text{Enc}(pk, m_b)$, called challenge ciphertext, is sent to the player.
4. **Guess**: Player C output guess $b' \in \{0, 1\}$.
5. **Win**: $\text{CPA}_C(\lambda) = \begin{cases} 1 & \text{if } b = b' \\ 0 & \text{if } b \neq b'. \end{cases}$

If $\mathrm{CPA}_C(\lambda) = 1$ then the adversary C guessed correctly which of their two chosen messages was encrypted, based only on observing the ciphertext. Notice that the game requires the player to choose messages of equal length in the 'choose' phase since the ciphertext length always leaks information about the length of the message, namely an upper bound on the message length.

Definition 21 (Semantic security (CPA))**.** An encryption scheme is semantically secure if, for all polynomial-size Boolean circuit families C,

$$|\Pr[\mathrm{CPA}_C(\lambda)] - \frac{1}{2}| = \mathrm{negl}(\lambda).$$

Semantic security means that there exists no algorithm in P/poly that can do more than negligibly better than guessing randomly in determining the message. Semantic security is equivalent to indistinguishability of encryptions (see [Gol04] for proof).

Definition 22 (Indistinguishability of encryptions)**.** An encryption scheme has indistinguishable encryptions if for any key $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda)$ and any two distinct messages m_1, m_2 of equal length, the ensembles $\{\mathrm{Enc}(pk, m_1)\}_{\lambda\in\mathbb{N}}$ and $\{\mathrm{Enc}(pk, m_2)\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable.

Usually, encryption schemes are required to be secure against a stronger type of attack, called chosen-ciphertext attack (CCA). There are two types of CCA attacks; adaptive (called CCA2) and non-adaptive (called CCA1). The CCA1 game is defined exactly like the CPA game but where the player also has oracle access to the decryption algorithm in the choose phase. In other words, the player can decrypt any ciphertext of their choice before submitting the two messages m_0 and m_1 to the challenger. The CCA2 game is the same as CCA1 except that the player also has oracle access to the decryption algorithm in the guess phase for every ciphertext except the challenge ciphertext. Security against CCA1 and CCA2 attacks are defined analogously to semantic security. Clearly, CCA2 security implies CCA1 security and CCA1 security implies semantic security.

As a consequence of its design, homomorphic encryption schemes cannot be CC2 secure. The reason is that the player can run the evaluate algorithm on the challenge ciphertext with any circuit of choice and then decrypt the evaluated ciphertext. More formally, consider any challenge ciphertext $\mathrm{Enc}(pk, m_b)$ and the identity circuit C. Is the identity circuit always in C?. Player runs $\mathrm{Eval}(pk, C, \mathrm{Enc}(pk, m_b))$, generating a valid evaluated ciphertext of $C(m_b)$. Player then queries decryption and yields $C(m_b) = m_b$.

Homomorphic encryption schemes allow the attacker to transform the ciphertext of a message m to a ciphertext to a message related to m by a known function. This property is called *malleability*.

One last security definition that will be relevant later is circular security

Definition 23 (Circular security)**.** A semantically secure homomorphic encryption scheme is circular secure if it is semantically secure even when the adversary is given encryptions of the secret key.

Remark. Circular security is not implied by semantic security because an adversary with a random access oracle cannot efficiently query encryptions of the secret key [Bra18].

Lattices

In this section, we assume every vector is a column vector and we denote a matrix $\mathbf{A}$ with boldface. By $\mathbf{A} \in \mathbb{F}^{n \times m}$ we mean $\mathbf{A}$ is a $n \times m$ matrix with entries in $\mathbb{F}$. Consider a basis $\mathbf{B} = \{b_1, \ldots, b_k\}$. A lattice with basis $\mathbf{B}$ is defined $L(\mathbf{B}) \overset{\text{def}}{=} \{\sum_{i=1}^{k} a_i b_i \mid a_i \in \mathbb{Z}\}$. For any integer $q \geq 2$, we define $\mathbb{Z}_q \overset{\text{def}}{=} (-\frac{q}{2}, \frac{q}{2}] \cap \mathbb{Z}$. For any tuple of integers x (e.g., integer or matrix), we define $[x]_q$ as the tuple of integers over $\mathbb{Z}_q$ such that each element is congruent mod q to the corresponding element in x. For example, $q = 3, x = (5, 1, -2, 6), [x]_q = (-1, 1, 1, 0)$.

LWE and RLWE

In 2005, Oded Regev introduced [Reg05] a natural problem; solve a system of modular noisy linear equations. The problem is called learning with errors (LWE) and in 2012, a ring based version called ring-LWE (RLWE) was introduced [LPR12]. Despite being easy to state, the LWE problem turns out to be hard even on average instances. For certain parameter choices, the LWE problem is reducible to extensively studied hard lattice based problems (e.g., GapSVP, SIVP) whereas RLWE is reducible to hard problems on ideal lattices (e.g., ideal-SVP, NTRU). Hardness of these problems are outside the scope of this thesis; we refer the reader to [Reg05; Pei08; Bra+13; LPR12] for more details on hardness results and [Pei15] for a good rundown on hard lattice based problems. Today, essentially all homomorphic encryption schemes are based on LWE and RLWE.

The parameters for LWE are the integers $n = n(\lambda)$ for the dimension, $q = q(n)$ for the global modulus, m for number of samples and a discrete error distribution measure $\chi = \chi(\lambda)$ over $\mathbb{Z}$.

Consider the space of $n \times m$ matrices $\mathbb{Z}_q^{n\times m}$ with uniform distribution, space of secrets $\mathbb{Z}_q^n$ with uniform distribution[1], space of errors $\mathbb{Z}^m$ with discrete distribution χ^m and the random variable $A_{n,q,\chi,m}$ defined on the direct product as follows

$$\begin{aligned} A_{n,q,\chi,m} \colon \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n \times \mathbb{Z}^m &\to \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m \cong \mathbb{Z}_q^{(n+1)\times m} \\ (\mathbf{A}, s, e) &\mapsto (\mathbf{A}, b^T \overset{\text{def}}{=} [s^T\mathbf{A} + e^T]_q) = \begin{bmatrix} \mathbf{A} \\ b^T \end{bmatrix} \end{aligned}$$

Definition 24 (LWE distribution)**.** The learning with errors distribution is defined as the distribution of $A_{n,q,\chi,m}$

$$\text{LWE}_{n,q,\chi,m} \overset{\text{def}}{=} \mathcal{L}(A_{n,q,\chi,m})$$

In words, sample a $n \times m$ matrix $\mathbf{A}$ with entries in $\mathbb{Z}_q$ at random, choose a uniformly random secret $s \leftarrow \mathbb{Z}_q^n$ and let e consist of m independent errors from the discrete distribution χ. Calculate b^T (i.e, $[s^T\mathbf{A} + e^T]_q$) and output the pair $(\mathbf{A}, b^T)$. The LWE distribution specifies the probability of sampling each pair.

There are two versions of the LWE problem; search-LWE and decision-LWE. Informally, take a sample $x \leftarrow \text{LWE}_{n,q,\chi,m}$ and consider the last row. The decision version is to decide whether the last row is a linear combination of the previous n rows or if it is uniformly random and the search version is to find the explicit linear combination assuming it exists.

Definition 25 (decision-LWE problem)**.** Let $\mathcal{U}_n$ be the uniform distribution on $\mathbb{Z}_q^{(n+1)\times m}$. Construct a PPT algorithm A such that

$$|\Pr[A(\mathcal{U}_n) = 1] - \Pr[A(\text{LWE}_{\text{n,q},\chi,\text{m}}) = 1]| > \text{negl}(n)$$

The decision-LWE hardness assumption is that $\mathcal{U}_n$ and $\text{LWE}_{n,q,\chi,m}$ are computationally indistinguishable (i.e., $\text{LWE}_{n,q,\chi,m}$ is pseudorandom).

Definition 26 (search-LWE problem)**.** Let $x = (\mathbf{A}, b^T) \leftarrow \text{LWE}_{n,q,\chi,m}$. Construct a PPT algorithm A such that

$$\Pr[A(x) = s] > \text{negl}(n)$$

The search-LWE hardness assumption is that $\Pr[A(x) = s] = \text{negl}(n)$. There exists

[1] The space of secrets can have distribution χ^n without loss of hardness. See [App+09].

a reduction from search-LWE to decision-LWE for $q = \text{poly}(n)$, meaning they are equivalently hard [Bra+13].

Regev's LWE based cryptosystem

In the same paper that the LWE problem was introduced, Regev also described how to construct a simple cryptosystem based on the LWE-assumption. In Regev's cryptosystem, each message is a bit. Given the security parameter λ, specify the parameters $n(\lambda), q(\lambda), \chi(\lambda), m(\lambda)$ for the system. For the purpose of security and concreteness, let $q = \text{poly}(n)$, χ be a discrete αq-bounded Gaussian distribution for $\alpha \in (0,1)$ and $m = n \log q$.

1. **Key generation**: Generate a uniformly random LWE secret $s' = (s_1, \ldots, s_n) \leftarrow \mathbb{Z}_q^n$ and let $s = (s_1, \ldots, s_n, -1) \in \mathbb{Z}_q^{n+1}$. Using s', generate the $(n+1) \times m$ matrix $\mathbf{A}' \overset{\text{def}}{=} (\mathbf{A}, b^T) \leftarrow \text{LWE}_{n,q,\chi,m}$. Let KeyGen($\lambda$) return $\mathbf{A}' \in \mathbb{Z}_q^{(n+1)\times m}$ as the public key and $s \in \mathbb{Z}_q^{n+1}$ as the secret key.

2. **Encryption**: Generate a random 'subset' vector $r \leftarrow \{0,1\}^m$. To encrypt a bit m, compute $c = \text{Enc}(\mathbf{A}', m) \overset{\text{def}}{=} [b \cdot \lfloor q/2 \rfloor \cdot (0, \ldots, 0, -1)^T + \mathbf{A}'r]_q \in \mathbb{Z}_q^{n+1}$.

3. **Decryption**: To decrypt, compute $z \overset{\text{def}}{=} [\langle s, c \rangle]_q$ and let Dec(s, c) be 0 if $|z| < q/4$ and 1 otherwise.

The idea behind the scheme is to embedd an encryption of a bit m in the last coordinate of the ciphertext c. To encrypt the bit, first consider a random subset of the m samples and then calculate their sum. Second, if and only if the bit is 1, add $\lfloor q/2 \rfloor$ to the last coordinate (i.e., last row). To see why decryption works, consider an encryption of 0. Then, $\text{Dec}(s, \text{Enc}(\mathbf{A}', 0)) = [\big\langle s, \mathbf{A}'r \big\rangle]_q = [\big\langle s\mathbf{A}', r \big\rangle]_q = [\langle e, r \rangle]_q$. If the bit is 1, then $\text{Dec}(s, \text{Enc}(\mathbf{A}', 1)) = [\big\langle s, (0, \ldots, 0, -\lfloor q/2 \rfloor)^T \big\rangle + \big\langle s, \mathbf{A}'r \big\rangle]_q = [\lfloor q/2 \rfloor + \langle e, r \rangle]_q$.

Correctness:

The security of the scheme is based on the hardness of two problems; subset-sum and LWE. An adversary that obtains r can easily decrypt a ciphertext; simply

Homomorphic properties:

Not bootstrappable:

RLWE

The LWE problem is conceptually easy to understand but suffers from expensive overhead. Each element in b is a perturbed linear combination of the secret s and the corresponsing column in $\mathbf{A}$, resulting in $O(n)$ operations. To get $m = n$ samples, the total number of operations is $O(n^2)$. In 2012, a more compact ring based learning with errors problem (RLWE) was introduced by Lyubashevsky, Peikert and Regev in [LPR12].

The parameters for RLWE are the integers $n = n(\lambda)$ for the dimension, $q = q(n)$ for the global modulus and a discrete error distribution measure $\chi = \chi(\lambda)$ over $\mathbb{Z}$.

Let the dimension $n = 2^k$ for some natural number k and consider the irreducible polynomial $f(x) = x^n + 1$. Define the polynomial ring $R_q \stackrel{\text{def}}{=} \mathbb{Z}_q[x]/\langle f(x)\rangle$. The ring R_q is viewed as the ring of polynomials with coefficients in $\mathbb{Z}_q$, having degree less than n. There is a natural correspondence between elements of R_q and $\mathbb{Z}_q^n$ where each coefficient corresponds to an element in the vector. Let the error space R'_q be R_q endowed with error distribution χ^n over the integer coefficients and consider the random variable $A'_{n,q,\chi}$ defined as follows

$$\begin{aligned} A'_{n,q,\chi} \colon R_q \times R'_q \times \mathbb{Z}^n &\to R_q^2 \\ (a, s, e) &\mapsto (a, b \stackrel{\text{def}}{=} [a \cdot s + e]_q) \end{aligned}$$

Definition 27 (RLWE distribution)**.** The learning with errors distribution is defined as the distribution of $A'_{n,q,\chi}$

$$\mathrm{RLWE}_{n,q,\chi} \stackrel{\text{def}}{=} \mathcal{L}(A'_{n,q,\chi})$$

In words, sample a uniformly random polynomial $a \in R_q$, a uniformly random secret $s \in R_q$ and a random error $e \in R_q$ from the discrete distribution χ^n. Calculate b (i.e, $[a \cdot s + e]_q$) and output the pair (a, b). The RLWE distribution specifies the probability of sampling each pair.

The decision-RLWE problem is to distinguish between the RLWE distribution and the uniform distribution on R_q^2 and the search problem is to find the secret s given a sample $(a, b) \leftarrow \mathrm{RLWE}_{n,q,\chi}$.

Definition 28 (decision-RLWE problem)**.** Let $\mathcal{U}_n$ be the uniform distribution on R_q^2.

Construct a PPT algorithm A such that

$$|\Pr[A(\mathcal{U}_n) = 1] - \Pr[A(\mathrm{RLWE_{n,q,\chi}}) = 1]| > \mathrm{negl}(n)$$

The decision-RLWE hardness assumption is that $\mathcal{U}_n$ and $\mathrm{RLWE}_{n,q,\chi}$ are computationally indistinguishable (i.e., $\mathrm{RLWE}_{n,q,\chi}$ is pseudorandom).

Definition 29 (search-RLWE problem)**.** Let $x = (a, b) \leftarrow \mathrm{RLWE}_{n,q,\chi}$. Construct a PPT algorithm A such that

$$\Pr[A(x) = s] > \mathrm{negl}(n)$$

The search-RLWE hardness assumption is that $\Pr[A(x) = s] = \mathrm{negl}(n)$.

Note that in LWE, the sample size m is embedded as a parameter while in RLWE, adversary generates $m = \mathrm{poly}(\lambda)$ samples from the $\mathrm{RLWE}_{n,q,\chi}$ themselves.

The main issue with basing schemes of the LWE problem is the inneficient key sizes. The key size is at least quadratic(why?) in the dimension n and hence the security parameter λ. The ring learning with errors problem (RLWE) was introduced by Lyubashevsky, Peikert and Regev in [LPR12] to address this issue. The RLWE problem is defined over a polynomial ring $\mathbb{Z}_q[x]/(x^n + 1)$ and the secret is a polynomial $s \in \mathbb{Z}_q[x]/(x^n + 1)$ with coefficients from $\mathbb{Z}_q$. The error is a polynomial $e \in \mathbb{Z}_q[x]/(x^n + 1)$ with coefficients from a discrete distribution χ. The RLWE distribution is defined as the distribution of the pair $(a, as + e)$ where a is a uniformly random polynomial in $\mathbb{Z}_q[x]/(x^n + 1)$. - LWE is inneficient.

History of fully homomorphic encryption

Began with rivest, adleman and Dertouzos in 1978 and the paper about Data Banks. They identified the use of delegating computation to a third party.

To perform homomorphic evaluation on arbitrary functions, a scheme needs to support unlimited number of multiplication and addition of ciphertexts. The problem was thus to find a scheme that both secure and also supports homomorphic addition and multiplication.

RSA supports multitplication,

Paillier supports addition.

First generation

Gentry scheme is based on ideal lattices. The construction considers a polynomial ring over the integers modulo an ideal generated by a cyclotomic polynomial. Then there exists an ideal with norm polynomially bounded with respect to the dimension such that the ideal generates a lattice (see [Gen09] for details). The security of Gentry's scheme was based on three hardness assumptions: sparse subset sum problem (SSSP), bounded-distance decoding problem (BDD) and the ideal shortest vector problem (ideal-SVP). The original decryption function in Gentry's scheme was not bootstrappable. He used a method called squashing to make it so. The idea was to include extra information in the public key which allowed for easier decryption but also introduced the SSSP hardness asumption.

The Gentry's scheme was improved by Smart and Vercauteren in [SV09] where they introduced batching of multiple plaintexts encrypted into a single ciphertext using the Chinese Remainder Theorem (ciphertext packing is the same idea where multiple ciphertexts are packed together). Gentry and Halevi implemented the scheme in [GH10], including the use of the batching technique and an optimized squashing technique to bring down the degree of the decryption polynomial from hundreds (estimated by Smart and Vercauteren) to 15. In Gentry's original scheme, the key generation algorithm was impractically slow. In their implementation, they reduced the asymptotic complexity to $\tilde{O}\left(n^{1.5}\right)$ for cyclotomic fields where the root of unity is a power of 2. Scholl and Smart extended the implementation to arbitrary cyclotomic fields in [SS11].

In 2010, Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan finalized a new scheme (DGHV scheme) that was based on multiplication and addition over the integers as opposed to over polynomial rings [Dij+09]. The hardness of the DGHV scheme was based on the approximate greatest common divisor problem (AGCD), and the SSSP due to squashing of the decryption circuit. Subsequent works optimised the DGHV scheme through modulus switching in [CNT11] and plaintext batching in [Kim+13] and [CLT13].

Second generation

The second generation began in 2011 with Brakerski and Vaikuntanathan [BV11]. Their paper introduced two main contributions. Their first contribution was to base the hardness of the proposed BV scheme on the well known LWE problem, which means using arbitrary lattices instead of ideal lattices. Their second contribution was the removal of squashing from previous schemes (Gentry and Halevi, independently, also managed to remove squashing in [GH11]). This meant that the BV scheme did not require the SSSP hardness assumption. As an optimization technique, the paper was the first to introduce what would later be called modulus switching. A downside of the BV scheme (and the rest of the schemes based on LWE in the second generation) was that an expensive step called re-linearization is nessesary to prevent homomorphic multiplication causing ciphertext length from growing exponentially. In the BV scheme, homomorphic multiplication corresponds to a tensor product, changing the structure of the ciphertext. In order to re-linearize the result, the scheme uses key-switching. This requires applying a large re-linearization matrix of size $\Omega(n^3)$ (embedded in the public key) to the tensorproduct.

In a follow up paper in 2011, Brakerski, Gentry and Vaikuntanathan introduced the BGV scheme that showed, for the first time, bootstrapping was not nessesary for a fully homomorphic encryption scheme. Their construction worked for both LWE and RLWE, and ironically used bootstrapping as an optimization technique. Brakerski later modified the LWE based BGV scheme by replacing the modulus switching technique with a scale invariant approach [Bra12] explain what he did. Fan and Vercauteren converted Brakerski's scale invariant scheme to the RLWE setting, resulting in the BFV scheme [FV12]. As a part of the second generation of homomorphic encryption, there were schemes based on the NTRU public key encryption scheme from 1998 [HPS98]. However, these schemes required parameter sizes larger than originally proposed due to vulnerabilities from subfield lattice attacks [ABD16].

Third generation

In 2013, Gentry, Sahai and Waters introduced a new scheme called the GSW scheme [GSW13]. The GSW scheme was also based on LWE, but unlike its predessors, it did not require the re-linearization step. This is because the scheme operates on matrices which has an inherent natural addition and multiplication operation. Since the relinearization matrix is no longer needed, the space complexity of GSW is quasi-quadratic as opposed to quasi-cubic for BGV and BFV. Furthermore, GSW also did not require bootstrapping to achieve FHE. A downside of the GSW scheme is the complexity of ciphertext matrix multiplication, yielding in slower multiplication that the ring based second generation schemes.

A defining trait of the third generation is the efficiency gain from bootstrapping. There were two main new faster bootstrapping algorithms. Jacob Alperin-Sheriff and Chris Peikert [AP13] introduced bootstrapping of non-packed ciphertexts in quasilinear time w.r.t the security parameter (which is important in the third generation as there is no packing) and Gama et al. [Gam+14] built on their work and introduced a new type of homomorphic gate. Léo Ducas and Daniele Micciancio developed a scheme called FHEW that used Alperin-Sheriff and Peikerts bootstrapping algorithm to allow for much faster bootstrapping using what is now called programmable bootstrapping [DM14]. Another paper introuced a scheme based on the LWE over the torus, meaning a stricter security assumption, called TFHE [Chi+18] which allowed for bootstrapping in 0.1 milliseconds after subsequent optimizations [MP20].

Fourth generation

The fourth generation of homomorphic encryption schemes began in 2016 when Jung Hee Cheon, Andrey Kim, Miran Kim and Yongsoo Song introduced a new RLWE based leveled FHE scheme called CKKS [Che+16] which they subsequently turned into a pure FHE scheme through bootstrapping in [Che+18]. CKKS is based on approximate arithmetic, allowing for computation on real and complex numbers. To achieve this, a vector of numbers (real or complex) are encoded using rounding as a integral plaintext polynomial in a cyclotomic polynomial ring. This is a fundamentally different approach as the scheme operates on an approximation of the message as opposed to the real message. The CKKS scheme is useful for applications concerning floating point numbers, such as machine learning and neural networks. Subsequent works focused on optimizing the bootstrapping and ciphertext packing techniques.

In 2020, Baiyu Li and Daniele Micciancio showed that the CKKS scheme was vulnerable to passive attacks with respect to IND-CPA security [LM20]. To mitigate this issue, a new security notion called IND-CPA+ was introduced. IND-CPA+ and IND-CCA1 differ in that the adversary is only allowed to query decryption on evaluated ciphertexts, as opposed to arbitrary ciphertexts. Li and Micciancio showed that IND-CPA+ is eequivalent to IND-CPA for exact encryption schemes but strictly stronger for approximate schemes like CKKS.

Noise management

The main problem with homomorphic encryption schemes is the noise. Noise is introduced in the ciphertexts during the encryption process and when the ciphertexts undergo homomorphic operations, the noise grows. After sufficiently many operations, the noise grows to the point where the decryption of the evaluated ciphertext fails. In order to reach FHE, it is necessary to control the noise to allow for sufficient number of operations. Noise management is the process of controlling the noise during homomorphic evaluation. In his 2009 seminal PHD thesis [Gen09], Craig Gentry showed that FHE was possible by using a noise management technique called Bootstrapping. Bootstrapping is an algorithm that transforms a possibly noisy ciphertext into a correct evaluated ciphertext with little noise by using an encryption of the secret key to decrypt the noisy ciphertext homomorphically.

Evaluated ciphertexts vs fresh ciphertexts

Recall that in Definition 17, the decryption of an evaluated ciphertext is only required to be correct if the ciphertext is a fresh ciphertext. Explain Why we care. The question is then if it is possible to pass evaluated ciphertext to the Eval algorithm instead of fresh ones.

A HE scheme is said to be i-hop if the Eval algorithm can be called on its own output up to i times. A HE scheme that satisfies the definition is 0-hop, also called *single-hop*, and multi-hop if it is i-hop for all i.

Note on hard-wiring ciphertexts in circuits

Not nessesary. Gentry did not hardwire, and instead he encrypted messages twice, passed the double encrypted message as inout where the inner layer was decrypted homomorphically. In practice, the server sees the ciphertext and constructs the dec function for that ciphertext specifically, designed to only take the secret key as input.

Key-Switching

This section is based on [Bra18].

As a natural segway into bootstrapping, we first introduce a related but different concept called key-switching. Since HE schemes are designed with the constraint of supporting homomorphic operations, they are often inefficient. Therefore, it would be desirable to use a more efficient non-homomorphic scheme to encrypt the large messages, but still retaining the homomorphic property. Key-switching is a technique that allows for homomorphic computation on ciphertexts encrypted under a non-homomorphic scheme. In particular, it allows for transforming a ciphertext under a non-HE scheme to a corresponding ciphertext under a HE scheme. The idea is to encrypt the much shorter secret key of the non-HE scheme under the public key of the HE scheme and hard-wire the ciphertexts into the function of interest.

Let (H.KeyGen, H.Enc, H.Dec, Eval) be a homomorphic encryption scheme and (KeyGen, Enc, Dec) be a non-homomorphic encryption scheme. Let $(hpk, hsk) \leftarrow \mathrm{H.\,KeyGen}(\lambda)$ and $(pk, sk) \leftarrow \mathrm{KeyGen}(\lambda)$. Consider computable function C, the vector of ciphertexts $c \leftarrow \mathrm{Enc}(pk, m)$ and an encrypted secret key $sk' \leftarrow \mathrm{H.\,Enc}(hpk, sk)$. Define $\hat{C}_c(\cdot) \overset{\mathrm{def}}{=} C(\mathrm{Dec}(\cdot, c))$. $C(m)$ can be computed homomorphically by decrypting $\mathrm{Eval}(hpk, \hat{C}_c, sk')$. Indeed, this is a correct encryption of $C(m)$ since

$$\mathrm{Dec}(hsk, \mathrm{Eval}(hpk, \hat{C}_c, sk')) = \hat{C}_c(sk) = C(\mathrm{Dec}(sk, c)) = C(m)$$

We have shown that it is possible to use a non-homomorphic encryption scheme to encrypt the message and still perform homomorphic computation on it. Key-switching transforms a ciphertext encrypted under a non-HE scheme to an evaluated ciphertext encrypted under a HE scheme.

The underlying assumption is that the HE scheme can evaluate $\hat{C}_c$, meaning it can first run the decryption algorithm and then compute the desired function C to generate a valid evaluated ciphertext. Even under the assumption that the decryption algorithm is simple enough to be evaluated by the scheme, a general C consists of several multiplication and addition gates, meaning it is unlikely that the scheme can evaluate $\hat{C}_c$. However, if it is possible to split C into smaller components, $C = C^m \circ \cdots \circ C^1$, and evaluate each component separately, $c_i = \text{Eval}(hpk, \hat{C}^i_{c_{i-1}}, sk')$, where $c_0 \leftarrow Enc(pk, m)$, computing C homomorphically is achievable assuming $\hat{C}^i_{c_{i-1}}$ is permissible for all i. However, as the construction currently stands, further computation on the ciphertext is not possible. To see why, consider $c_1 = \text{Eval}(hpk, \hat{C}^1_{c_0}, sk')$ and let $c_2 = \text{Eval}(hpk, \hat{C}^2_{c_1}, sk')$. We hope decrypting c_2 yields $(C^2 \circ C^1)(m)$, but $\text{Dec}(hsk, c_2) = \text{Dec}(hsk, \text{Eval}(hpk, \hat{C}^2_{c_1}, sk')) = C^2(\text{Dec}(sk, c_1))$. However, since c_1 is encrypted under the HE scheme, the non-homomorphic decryption algorithm applied to c_1 does not make sense and decryption fails.

Bootstrapping

Key-switching is a nice optimization technique for encrypting messages faster, but it does not allow for further computation on the ciphertext. The requirement is that the decryption algorithm and the secret key originate from the HE scheme. We therefore only consider the HE scheme. Let (KeyGen, Enc, Dec, Eval) be a HE scheme, $(pk, sk) \leftarrow \text{KeyGen}(\lambda)$ and $sk' \leftarrow \text{Enc}(pk, sk)$. Since this construction encrypts the secret key using its own public key, circular security is assumed (this assumption is discussed further below). Under this construction, decryption of c_1 is still correct as $\text{Dec}(sk, c_1) = C^1_{c_0}(sk) = C^1(\text{Dec}(sk, c_0)) = C^1(m)$, but the difference is that decryption of c_i also works since, by induction,

$$\text{Dec}(hsk, \text{Eval}(hpk, \hat{C}^i_{c_{i-1}}, sk')) = \hat{C}^i_{c_{i-1}}(sk) = C^i(\text{Dec}(sk, c_{i-1})) = C^i \circ C^{i-1} \circ \cdots \circ C^1(m)$$

In essence, we have constructed a method to evaluate C as follows: The first step is to evaluate a component of C on the encryption of the secret key to obtain an evaluated ciphertext. The second step is to generate a circuit which is the composition of the decryption circuit for the previous evaluated ciphertext and the consecutive component of C. The third step is to evaluate the generated circuit on the same ciphertext, namely the encryption of the secret key. Repeating this process ultimately yields c_m which decrypts to $C(m)$. Since each evaluated ciphertext $c_1, \dots, c_m$ is an evaluation of a fresh ciphertext, sk', the decryption is guaranteed to work.

Splitting the desired function C into multiple components $C^1, \dots, C^m$ is the key insight into constructing FHE. Consider the simplest case; each component function is either one multiplication gate or one addition gate. In other words, for a vector of ciphertexts $c = \langle c_1, c_2 \rangle$, define a multiplicative component circuit of C as $C^i(c) = c_1 \times c_2$, $\hat{C}^i_c(\cdot) = C^i(\mathrm{Dec}(\cdot, c)) = C^i(\langle \mathrm{Dec}(\cdot, c_1), \mathrm{Dec}(\cdot, c_2) \rangle) = \mathrm{Dec}(\cdot, c_1) \times \mathrm{Dec}(\cdot, c_2)$ and where the addition case is analogous. Note that $\hat{C}^i_c(sk) = m_1 \times m_2$. It turns out that if the scheme can evaluate just these two types of circuits, then it can evaluate every circuit. This is the idea behind bootstrapping.

Definition 30 (Bootstrappable encryption scheme)**.** Let $\mathcal{E}$ be a $\mathcal{C}$-homomorphic encryption scheme. Consider the following *augmented decryption circuits* for $\mathcal{E}$:

$$B^{(m)}_{c_1,c_2}(\cdot) \overset{\text{def}}{=} \mathrm{Dec}(\cdot, c_1) \times \mathrm{Dec}(\cdot, c_2)$$
$$B^{(a)}_{c_1,c_2}(\cdot) \overset{\text{def}}{=} \mathrm{Dec}(\cdot, c_1) + \mathrm{Dec}(\cdot, c_2)$$

$\mathcal{E}$ is *bootstrappable* if

$$\{B^{(m)}_{c_1,c_2}(\cdot), B^{(a)}_{c_1,c_2}(\cdot) \mid c_1, c_2 \in \mathcal{Y}\} \subset \mathcal{C}$$

The augmented decryption circuits have the ciphertexts hard-wired into its description and takes as input an encryption of the secret key. A bootstrappable scheme correctly evaluates the set of all augmented decryption circuits. In particular, it correctly evaluates its own decryption circuit by letting c_2 be an encryption of the multiplicative or additive identity respectively.

Theorem 2 (Gentrys bootstrapping theorem, simplified by Vaikuntanathan [Gen09; Vai11])**.** *Any bootstrappable scheme is leveled-FHE. Furthermore, if circular security holds, it is pure FHE.*

Remark. Bootstrapping is sufficient for leveled-FHE, but not necessary. See [BGV14] for a leveled-FHE scheme that does not require bootstrapping.

Proof. test □

We first introduce notation and then explain bootstrapping further. Let m' denote encrypted message m and sk' denote encrypted secret key sk. If a scheme assumes circular security, then one valid key pair (pk, sk) is sufficient for arbitrarily many subroutine calls where every encryption is made using pk. However, if circular security is not assumed, every bootstrapping subroutine call requires a new valid key pair. Consider a sequence of independently sampled valid key pairs $\{(pk_i, sk_i)\}_{i=0,1,\dots}$. Let $m^{(k)}$ and $sk_i^{(k)}$ denote valid encryptions under public key pk_k, meaning $\text{Dec}(sk_k, m^{(k)}) = m$ and $\text{Dec}(sk_k, sk_i^{(k)}) = sk_i$. The first bootstrapping subroutine call is done using $i = 0$. In the bootstrapping algorithm we set $k = i+1$ so that the encryption of every secret key made under the next public key, meaning the circular security assumption is avoided.

Showing how a bootstrapping subroutine works

Consider a circular secure bootstrappable homomorphic encryption scheme $\mathcal{E}$ and assume we need the homomorphic multiplication[1] of two ciphertexts m_1', m_2' encrypted under pk, but that the noise of the resulting evaluated ciphertext would cause decryption to fail. Consider instead the less noisy ciphertext $m_{1,2}' \overset{\text{def}}{=} B^{(m)}_{m_1',m_2'}(sk')$. In other words, encrypt the secret key and pass it as input to the circuit.

Indeed, $m_{1,2}'$ is a valid ciphertext of $m_1 \times m_2$ since

$$\begin{aligned}\text{Dec}(sk, m_{1,2}') &= \text{Dec}(sk, B^{(m)}_{m_1',m_2'}(sk'))\\ &= B^{(m)}_{m_1',m_2'}(sk)\\ &= \text{Dec}(sk, m_1') \times \text{Dec}(sk, m_2')\\ &= m_1 \times m_2\end{aligned}$$

If circular security is not assumed, the process becomes more complicated. For bootstrapping round i, where $i = 0$ represents the first round, let $m_{1,2}^{(i+1)} \overset{\text{def}}{=}$

[1] The case for addition of ciphertexts works the exact same way. We focus on multiplication here since addition usually only increases the noise slightly.

$B^{(m)}_{m_1^{(i)},m_2^{(i)}}(sk_i^{(i+1)})$. Correctness follows from

$$\begin{aligned}
\mathrm{Dec}(sk_{i+1}, m_{1,2}^{(i+1)}) &= \mathrm{Dec}(sk_{i+1}, B^{(m)}_{m_1^{(i)},m_2^{(i)}}(sk_i^{(i+1)})) \\
&= B^{(m)}_{m_1^{(i)},m_2^{(i)}}(sk_i) \\
&= \mathrm{Dec}(sk_i, m_1^{(i)}) \times \mathrm{Dec}(sk_i, m_2^{(i)}) \\
&= m_1 \times m_2
\end{aligned}$$

Note that both $m'_{1,2}$ and $m'_1 \times m'_2$ decrypts to $m_1 \times m_2$. The idea behind bootstrapping is that $m'_{1,2}$ has less noise.

In practice, we don't use augmented decryption circuits but instead we just refresh the ciphertext. In this case, we only need the dec func to be permitted.

Bibliography

[Adl78] Adleman, Leonard M. “Two theorems on random polynomial time”. In: *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)* (1978), pp. 75–83.

[ABD16] Albrecht, Martin, Bai, Shi, and Ducas, Léo. *A subfield lattice attack on overstretched NTRU assumptions: Cryptanalysis of some FHE and Graded Encoding Schemes*. Cryptology ePrint Archive, Paper 2016/127. https://eprint.iacr.org/2016/127. 2016. URL: https://eprint.iacr.org/2016/127.

[AP13] Alperin-Sheriff, Jacob and Peikert, Chris. *Practical Bootstrapping in Quasilinear Time*. Cryptology ePrint Archive, Paper 2013/372. https://eprint.iacr.org/2013/372. 2013. URL: https://eprint.iacr.org/2013/372.

[App+09] Applebaum, Benny, Cash, David, Peikert, Chris, and Sahai, Amit. “Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems”. In: *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference*. Vol. 5677. Lecture Notes in Computer Science. Springer, 2009, pp. 595–618. DOI: 10.1007/978-3-642-03356-8_35. URL: https://www.iacr.org/archive/crypto2009/56770585/56770585.pdf.

[AB09] Arora, Sanjeev and Barak, Boaz. *Computational Complexity: A Modern Approach*. 1st. New York, NY, USA: Cambridge University Press, 2009.

[Bra12] Brakerski, Zvika. *Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP*. Cryptology ePrint Archive, Paper 2012/078. https://eprint.iacr.org/2012/078. 2012. URL: https://eprint.iacr.org/2012/078.

[Bra18] Brakerski, Zvika. “Fundamentals of Fully Homomorphic Encryption - A Survey”. In: *Electronic Colloquium on Computational Complexity* 125 (2018).

[BGV14] Brakerski, Zvika, Gentry, Craig, and Vaikuntanathan, Vinod. “(Leveled) Fully Homomorphic Encryption Without Bootstrapping”. In: *ACM Transactions on Computation Theory (TOCT)* 6.3 (2014). Preliminary version in ITCS 2012, p. 13.

[Bra+13] Brakerski, Zvika, Langlois, Adeline, Peikert, Chris, Regev, Oded, and Stehlé, Damien. *Classical Hardness of Learning with Errors.* 2013. arXiv: `1306.0281 [cs.CC]`.

[BV11] Brakerski, Zvika and Vaikuntanathan, Vinod. *Efficient Fully Homomorphic Encryption from (Standard) LWE.* Cryptology ePrint Archive, Paper 2011/344. `https://eprint.iacr.org/2011/344`. 2011. URL: `https://eprint.iacr.org/2011/344`.

[Che+18] Cheon, Jung Hee, Han, Kyoohyung, Kim, Andrey, Kim, Miran, and Song, Yongsoo. *Bootstrapping for Approximate Homomorphic Encryption.* Cryptology ePrint Archive, Paper 2018/153. `https://eprint.iacr.org/2018/153`. 2018. URL: `https://eprint.iacr.org/2018/153`.

[Che+16] Cheon, Jung Hee, Kim, Andrey, Kim, Miran, and Song, Yongsoo. *Homomorphic Encryption for Arithmetic of Approximate Numbers.* Cryptology ePrint Archive, Paper 2016/421. `https://eprint.iacr.org/2016/421`. 2016. URL: `https://eprint.iacr.org/2016/421`.

[Chi+18] Chillotti, Ilaria, Gama, Nicolas, Georgieva, Mariya, and Izabachène, Malika. *TFHE: Fast Fully Homomorphic Encryption over the Torus.* Cryptology ePrint Archive, Paper 2018/421. `https://eprint.iacr.org/2018/421`. 2018. URL: `https://eprint.iacr.org/2018/421`.

[CNT11] Coron, Jean-Sebastien, Naccache, David, and Tibouchi, Mehdi. *Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the Integers.* Cryptology ePrint Archive, Paper 2011/440. `https://eprint.iacr.org/2011/440`. 2011. URL: `https://eprint.iacr.org/2011/440`.

[CLT13] Coron, Jean-Sébastien, Lepoint, Tancrède, and Tibouchi, Mehdi. *Batch Fully Homomorphic Encryption over the Integers.* Cryptology ePrint Archive, Paper 2013/036. https://eprint.iacr.org/2013/036. 2013. URL: https://eprint.iacr.org/2013/036.

[Dij+09] Dijk, Marten van, Gentry, Craig, Halevi, Shai, and Vaikuntanathan, Vinod. *Fully Homomorphic Encryption over the Integers.* Cryptology ePrint Archive, Paper 2009/616. https://eprint.iacr.org/2009/616. 2009. URL: https://eprint.iacr.org/2009/616.

[DM14] Ducas, Léo and Micciancio, Daniele. *FHEW: Bootstrapping Homomorphic Encryption in less than a second.* Cryptology ePrint Archive, Paper 2014/816. https://eprint.iacr.org/2014/816. 2014. URL: https://eprint.iacr.org/2014/816.

[FV12] Fan, Junfeng and Vercauteren, Frederik. *Somewhat Practical Fully Homomorphic Encryption.* Cryptology ePrint Archive, Paper 2012/144. https://eprint.iacr.org/2012/144. 2012. URL: https://eprint.iacr.org/2012/144.

[Gam+14] Gama, Nicolas, Izabachene, Malika, Nguyen, Phong Q., and Xie, Xiang. *Structural Lattice Reduction: Generalized Worst-Case to Average-Case Reductions and Homomorphic Cryptosystems.* Cryptology ePrint Archive, Paper 2014/283. https://eprint.iacr.org/2014/283. 2014. URL: https://eprint.iacr.org/2014/283.

[Gen09] Gentry, Craig. "A Fully Homomorphic Encryption Scheme". AAI3382729. PhD thesis. Stanford, CA, USA: Stanford University, 2009.

[GH11] Gentry, Craig and Halevi, Shai. *Fully Homomorphic Encryption without Squashing Using Depth-3 Arithmetic Circuits.* Cryptology ePrint Archive, Paper 2011/279. https://eprint.iacr.org/2011/279. 2011. URL: https://eprint.iacr.org/2011/279.

[GH10] Gentry, Craig and Halevi, Shai. *Implementing Gentry's Fully-Homomorphic Encryption Scheme.* Cryptology ePrint Archive, Paper 2010/520. https://eprint.iacr.org/2010/520. 2010. URL: https://eprint.iacr.org/2010/520.

[GSW13] Gentry, Craig, Sahai, Amit, and Waters, Brent. *Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based.* Cryptology ePrint Archive, Paper 2013/340. https://eprint.iacr.org/2013/340. 2013. URL: https://eprint.iacr.org/2013/340.

[Gol08] Goldreich, Oded. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, 2008. DOI: 10.1017/CBO9780511804106.

[Gol01] Goldreich, Oded. *Foundations of Cryptography.* Vol. I. Cambridge University Press, 2001.

[Gol04] Goldreich, Oded. *Foundations of Cryptography.* Vol. II. Cambridge University Press, 2004.

[HPS98] Hoffstein, Jeffrey, Pipher, Jill, and Silverman, Joseph H. “NTRU: A ring-based public key cryptosystem”. In: *Algorithmic Number Theory.* Ed. by Joe P. Buhler. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 267–288. ISBN: 978-3-540-69113-6.

[Kal21] Kallenberg, Olav. *Probability Theory and Stochastic Modelling.* 3rd ed. Probability Theory and Stochastic Modelling. Cham, Switzerland: Springer Nature Switzerland AG, 2021. ISBN: 978-3-030-61870-4. DOI: 10.1007/978-3-030-61871-1.

[Kim+13] Kim, Jinsu, Lee, Moon Sung, Yun, Aaram, and Cheon, Jung Hee. *CRT-based Fully Homomorphic Encryption over the Integers.* Cryptology ePrint Archive, Paper 2013/057. https://eprint.iacr.org/2013/057. 2013. URL: https://eprint.iacr.org/2013/057.

[LM20] Li, Baiyu and Micciancio, Daniele. *On the Security of Homomorphic Encryption on Approximate Numbers.* Cryptology ePrint Archive, Paper 2020/1533. https://eprint.iacr.org/2020/1533. 2020. URL: https://eprint.iacr.org/2020/1533.

[LPR12] Lyubashevsky, Vadim, Peikert, Chris, and Regev, Oded. *On Ideal Lattices and Learning with Errors Over Rings.* Cryptology ePrint Archive, Paper 2012/230. https://eprint.iacr.org/2012/230. 2012. URL: https://eprint.iacr.org/2012/230.

[MP20] Micciancio, Daniele and Polyakov, Yuriy. *Bootstrapping in FHEW-like Cryptosystems*. Cryptology ePrint Archive, Paper 2020/086. https://eprint.iacr.org/2020/086. 2020. URL: https://eprint.iacr.org/2020/086.

[MR07] Micciancio, Daniele and Regev, Oded. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: *SIAM Journal on Computing* 37.1 (2007), pp. 267–302. DOI: 10.1137/S0097539705447360. eprint: https://doi.org/10.1137/S0097539705447360. URL: https://doi.org/10.1137/S0097539705447360.

[MF21] Mittelbach, Arno and Fischlin, Marc. *The Theory of Hash Functions and Random Oracles*. Vol. I. Springer Cham, 2021.

[Pei15] Peikert, Chris. *A Decade of Lattice Cryptography*. Cryptology ePrint Archive, Paper 2015/939. https://eprint.iacr.org/2015/939. 2015. URL: https://eprint.iacr.org/2015/939.

[Pei08] Peikert, Chris. *Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem*. Cryptology ePrint Archive, Paper 2008/481. https://eprint.iacr.org/2008/481. 2008. URL: https://eprint.iacr.org/2008/481.

[Reg05] Regev, Oded. "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography". In: STOC '05. Baltimore, MD, USA: Association for Computing Machinery, 2005, pp. 84–93. ISBN: 1581139608. DOI: 10.1145/1060590.1060603. URL: https://doi.org/10.1145/1060590.1060603.

[SS11] Scholl, P. and Smart, N. P. *Improved Key Generation For Gentry's Fully Homomorphic Encryption Scheme*. Cryptology ePrint Archive, Paper 2011/471. https://eprint.iacr.org/2011/471. 2011. URL: https://eprint.iacr.org/2011/471.

[SV09] Smart, N. P. and Vercauteren, F. *Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes*. Cryptology ePrint Archive, Paper 2009/571. https://eprint.iacr.org/2009/571. 2009. URL: https://eprint.iacr.org/2009/571.

[Vai11] Vaikuntanathan, Vinod. “Computing Blindfolded: New Developments in Fully Homomorphic Encryption”. In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*. University of Toronto. IEEE, 2011. DOI: 10.1109/FOCS.2011.98.

Contents

First Appendix

This is only slightly related to the rest of the report

Second Appendix

this is the information

www.kth.se