Homomorphic Encryption

KTH Thesis Report

Daniel Sanaee

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

Kandidat examensarbete, ...

Acknowledgements

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

Acronyms

Contents

Chapter 1

Theory

INTRODUCTION

1.1 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 polynomial $C \colon \mathbb{F}^n \to \mathbb{F}^m$.

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

[1] A Boolean circuit is an arithmetic curcuit. Conversely, any Boolean circuit can be simulated by

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 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 henceforth assume every Turing machines halts. 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 $r, m = \text{poly}(n)$.

Nowhere in the HE definitions below do we actually use circuit families?

For any input x with length n, $C(x) \stackrel{\text{def}}{=} C_n(x)$. 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.

Definition 3 (Universal set of gates)**.** Any set of gates that can implement any effectively computable function.

In this paper, we will use the arithmetic gates of multiplication and addition. This is equivalent to the binary gates XOR and AND.

Complexity classes

Definition 4 (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 5 (Complexity Class BPP)**.** BPP is the set of languages $\mathscr{L}$ such that there exists a probabilistic polynomial-time (PPT) Turing machine M satisfying

converting every gate to XOR and AND gates and using $\text{XOR}(a, b) = a + b - 2 * a * b$, $\text{AND}(a, b) = a * b$.

[2]The reason for this alternative model is to assume adversaries are computationally "stronger". See Theorem **??**.

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

Definition 6 (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:

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 the 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 7 (Uniform circuit family [AB09])**.** 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.[4]

1.2 Probability Theory

Behöver jag referera i definitionerna?

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

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 ensemble $\{Enc(pk, m)\}_{pk \in \mathbb{N}}$

Definition 9 (Statistical distance of ensembles)**.** 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

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

Definition 10. 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 $f = \text{negl}(n)$.

A function that is not negligible is called noticible.

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

[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

[4] Note however that set of languages solvable by a Turing machine that takes polynomial size advice is equivalent to P/poly.

Definition 12 (Statistically indistinguishable)**.** Two ensembles 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 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 random variable the samples are generated from. The formal definition is as follows:

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

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

The randomness is taken over the outcomes of the random variables X_n and Y_n.

Here the outputs 1 are immaterial. The point is that, for every adversary C guessing the samples are from one of the random variables (1 in this case represents from Y), the probability of being correct is, up to negligible difference, 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 random variables in the ensembles goes to zero fast (faster than the inverse of all polynomials).

In cryptography, a probability ensembles corresponds to a PPT algorithm where each index represents an input size. We therefore only consider sample spaces with polynomial cardinality in the index. In other words, for every ensemble $X = (X_n \colon \Omega_n \to S_n)_{n\in\mathbb{N}}$ there exists a polynomial p such that for every n, $|S_n| < p(n)$.

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

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

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.

Most (if not all) homomorphic encryption schemes are based on noise. As we will see in Subsection **??**, 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 14 (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, $\chi(A) = \sum_{x_i \in A} \Pr[X = x] \quad A \in \mathcal{S}$

Remark. Two different random variables can have the same distribution and one random variable can have two different distributions. See [Shr04] for a formal and excellent discussion.

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 $x \leftarrow X$

to denote sampling the outcome x assuming X is uniformly distributed (probability of all singletons are equal) and we write $x \leftarrow \chi$ for sampling the outcome x assuming χ is the distribution of X.

Definition 15 (Statistical distance of distributions)**.** Let χ_1, χ_2 be two discrete distributions (i.e., probability measures) on the measurable space $(S, \mathcal{S})$. The statistical distance is defined as

$$\Delta(\chi_1, \chi_2) \stackrel{\text{def}}{=} \frac{1}{2} \sum_{x \in S} |\chi_1(\{x\}) - \chi_2(\{x\})|$$

Remark. This definition assumes every singleton is measurable. A more general definition for continuous measure is $\Delta(\chi_1, \chi_2) \stackrel{\text{def}}{=} \sup_{A \in \mathcal{S}} |\chi_1(A) - \chi_2(A)|$

Add a part about discrete gaussian distribution

1.3 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 $\mathcal{E} =$ (KeyGen, Enc, Dec) is a triple of algorithms satisfying the following:

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

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 schemes. These schemes include a fourth algorithm, Eval, called the evaluation algorithm which is used by the server

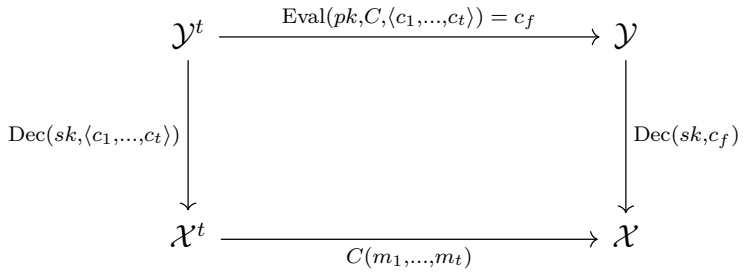

Figure 1: The decryption homomorphism. The path through $\mathcal{Y}$ represents computing before decrypting. The path through $\mathcal{X}^t$ represents decrypting before computing.

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:

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

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

The arithmetic circuits are polynomials multiplying and adding inputs. In the case of Boolean circuits, the operations are AND and XOR instead. The evaluation algorithm calculates the corresponding polynomial for the given circuit on the inputs. It makes sense to construct new algorithms that calculate each operation separately and, when stacked, are used together to calculate the final result. Consider therefore the algorithms $\mathrm{EvalMult}(c_1, c_2)$ and $\mathrm{EvalAdd}(c_1, c_2)$ outputting the product and the sum of ciphertexts respectively.

The evaluation function runs the ciphertexts through the permissible circuit and returns the output of the circuit. The requirement is that decrypting the resulting ciphertext yields the same result as the plaintexts running through the circuit. 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*. 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., $c_f \stackrel{\text{def}}{=} \mathrm{Eval}(pk, f, c)$). Similarly, for the vector m of plaintexts, we denote the evaluated plaintexts m_f (i.e., $m_f \stackrel{\text{def}}{=} f(m)$). Thus, the condition for the Eval algorithm can be written $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda) \Rightarrow \mathrm{Dec}(sk, c_f) = m_f$.

In a homomorphic encryption scheme, the decryption function is a homomorphism. Consider a valid key pair (sk, pk) hard coded into the decryption and evaluate functions respectively, plaintext ciphertext pairs $(m_1, c_1 = \text{Enc}_{pk}(m_1))$, $(m_2, c_2 = \text{Enc}_{pk}(m_2))$ and circuits $C_+(m_1, m_2) \stackrel{\text{def}}{=} m_1 + m_2$ and $C_\times(m_1, m_2) \stackrel{\text{def}}{=} m_1 \times m_2$ that can be evaluated by the scheme.

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

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 for $\mathcal{C}$ 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 complexity (i.e., for any constant b, there exists a circuit in the set with description length greater than b). This 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 complexity. 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 user.

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:

- 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 $\mathrm{CPA}_A(\lambda)$ as follows:

1. **Setup**: Challenger samples $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda)$ and sends player pk.
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 $c \stackrel{\text{def}}{=} \mathrm{Enc}(pk, m_b)$ is sent to the player.
4. **Guess**: Player C output guess $b' \in \{0, 1\}$.
5. **Win**: $\mathrm{CPA}_C(\lambda) = \begin{cases} 1 & \text{if } b = b' \\ 0 & \text{if } b \neq b'. \end{cases}$

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.

Although semantic security suggests adversary cannot distinguish plaintexts by only observing ciphertexts, it does not mean that ciphertexts give no information about messages. In fact, the ciphertexts in a homomorphic encryption scheme must leak information in some sense. To see why, for any message m and any circuit C that the

homomorphic encryption scheme evaluates, $\text{Eval}(pk, C, \text{Enc}(pk, m))$ is a valid ciphertext of $C(m)$, meaning that the ciphertext of the message leaks information about valid encryptions of messages directly related to m. In particular, the adversary can sample polynomially many valid encryptions of the message itself. However, if semantic security holds, the ensemble of ciphertexts generated by the message is computationally indistinguishable from any other ensemble of ciphertexts. We say that homomorphic encryption schemes are *malleable*.

One last security definition that will be relevant in the next section is circular security circular security - present the alternative (a long chain of keys where the next public key encrypts the previous public key. See TR18 Remark: Circular security is not implied by indistinguishability because the adversary cannot generate encryptions of the secret key. (Braverski)

Bootstrapping

Most (if not all) homomorphic encryption scheme is based on introducing noise to ciphertexts in the encryption process, which is useful for protecting against linear algebra attacks (see Section **??**). The evaluation process operates on ciphertexts using the homomorphic properties of the scheme to yield new ciphertexts corresponding to the same operations on the underlying plaintexts. Unfortunately, each operation increases the noise of the resulting ciphertexts. For a leveled FHE scheme with depth parameter set to L, ciphertexts with any more than L operations are not guaranteed to decrypt correctly. Bootstrapping is a technique introduced by Craig Gentry in his seminal PHD thesis [Gen09], where he showed FHE is possible. Bootstrapping is an algorithm that produces a new valid ciphertext with less noise by encrypting the secret key and decrypting the ciphertext homomorphically. In short, the client gives the server an encryption of the secret key and the server runs the decryption algorithm on the messages using the encrypted secret key.

Does this analogy even make sense? There is no double encryption in my Bootstrapping algorithm? A helpful analogy is to consider a powerful river blocked by an old dam. The dam will not last much longer so it will be replaced. An intuitive approach is to first build a new dam placed after the old dam before demolishing it. In the case of bootstrapping, the old dam is the noisy ciphertext, the new dam is a fresh encryption and the demolition represents homomorphic decryption.

Definition 23 (Bootstrappable encryption scheme)**.** Let $\mathcal{E}$ be a $\mathcal{C}$-homomorphic encryption scheme with decryption function Dec. Consider the following *augmented decryption circuits* for $\mathcal{E}$:

$$B^{(mult)}_{c_1,c_2}(\cdot) \stackrel{\text{def}}{=} \text{Dec}(\cdot, c_1) \times \text{Dec}(\cdot, c_2)$$
$$B^{(add)}_{c_1,c_2}(\cdot) \stackrel{\text{def}}{=} \text{Dec}(\cdot, c_1) + \text{Dec}(\cdot, c_2)$$

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

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

Why do we hardcode the ciphetexts? To gain efficiency? The augmented decryption circuits have the ciphertexts hard-coded into its description and takes as input an element representing either the secret key or an encryption of a 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. There exist multiple techniques used for controlling noise growth.

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 $\{(pk_i, sk_i)\}_{i=0,1,\ldots}$. 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 the encryption of every secret key made under the 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[5] 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 $m'_{1,2} \stackrel{\text{def}}{=} B^{(mult)}_{m'_1,m'_2}(sk')$. In other words, encrypt the secret key and pass it as input to the circuit.

Pseudocode for bootstrapping algorithm here?

Correctness here? Indeed, $m'_{1,2}$ is a valid ciphertext of $m_1 \times m_2$ since

$$\begin{aligned}
\text{Dec}(sk, m'_{1,2}) &= \text{Dec}(sk, B^{(mult)}_{m'_1,m'_2}(sk')) \\
&= B^{(mult)}_{m'_1,m'_2}(sk) \\
&= \text{Dec}(sk, m'_1) \times \text{Dec}(sk, m'_2) \\
&= m_1 \times m_2
\end{aligned}$$

If circular security is not assumed, the process becomes more complicated. For bootstrapping round i, where $i = 0$ represents the first round, let $m^{(i+1)}_{1,2} \stackrel{\text{def}}{=} B^{(mult)}_{m^{(i)}_1,m^{(i)}_2}(sk^{(i+1)}_i)$. Correctness follows from

$$\begin{aligned}
\text{Dec}(sk_{i+1}, m^{(i+1)}_{1,2}) &= \text{Dec}(sk_{i+1}, B^{(mult)}_{m^{(i)}_1,m^{(i)}_2}(sk^{(i+1)}_i)) \\
&= B^{(mult)}_{m^{(i)}_1,m^{(i)}_2}(sk_i) \\
&= \text{Dec}(sk_i, m^{(i)}_1) \times \text{Dec}(sk_i, m^{(i)}_2) \\
&= m_1 \times m_2
\end{aligned}$$

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.

Explain why noise level of $m'_{1,2}$ is less than $m'_1 \times m'_2$

Showing here why gentry theorem is true Assume the homomorphic encryption scheme $\mathcal{E}$ can evaluate the bootstrapping circuits. Consider a

Formalize as algorithm pseudocode

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

1.4 Lattices

In this section, we assume every vector is a column vector and we denote a matrix $\mathbf{A}$ with boldface. By $\mathbf{A} \in \mathbb{F}n \times m$ 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 $\mathcal{L}(\mathbf{B}) \stackrel{\text{def}}{=} \{\sum_{i=1}^{k} a_i b_i \mid a_i \in \mathbb{Z}\}$. For any integer $q \geq 2$, we define $Z_q \stackrel{\text{def}}{=} (-\frac{q}{2}, \frac{q}{2}] \cap \mathbb{Z}$. For any tuple of integers x (e.g., integer or matrix), we define $[x]_q$ as the tuple of integers over 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

This section is based on [Hal17], [LNP22] and [Pei15].

The LWE problem is to solve a system of linear equations with a small noise added. The parameters for the LWE problem are the integers n for the dimension, $q = q(n)$ for the modulus, m for number of samples and a discrete error distribution measure χ on $\mathbb{Z}_q$.

The requirement is $q(n) = \Omega(n)$(Is this true?), $m = \Theta(n \log q)$ and $\Pr[|x| \geq aq \mid x \leftarrow \chi] = negl(n)$. I think this is requirement for hardness, not for the problem itself. This means that as the dimension grows and q also grows, each component of the error grows (assuming a doesnt decrease fast than q)

Consider the space of $n \times m$ matrices $Z_q^{n \times m}$ with uniform distribution μ, Z_q^m with discrete distribution χ^m, the probability space $(Z_q^{n \times m} \times Z_q^m, \mathcal{P}, \mu \otimes \chi^m)$ equipped with power set sigma algebra $\mathcal{P}$ and product measure $\mu \otimes \chi^m$, a uniformly random[6] $s \in \mathbb{Z}_q^n$ and the random variable $A_{s,n,q,\chi,m}$ defined on the probability space as follows

$$\begin{aligned} A_{s,n,q,\chi,m} \colon Z_q^{n \times m} \times Z_q^m &\to Z_q^{n \times m} \times Z_q^m \\ (\mathbf{A}, e) &\mapsto (\mathbf{A}, s^T \mathbf{A} + e^T) \end{aligned}$$

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

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

[6]Secret s can also be chosen with respect to distribution χ^n. See [App+09].

Remark. There are many LWE distributions, each characterised by the the parameters n, m, q, s, χ.

In words, first fix a secret s with entries in $\mathbb{Z}_q$ at random. Then sample a $n \times m$ matrix $\mathbf{A}$ with entries in $\mathbb{Z}_q$ at random and e consisting of m independent errors from the discrete distribution χ. Calculate $s^T\mathbf{A} + e^T$ and output the pair $(\mathbf{A}, s^T\mathbf{A} + e^T)$. The $\text{LWE}_{s,m,\chi}$ distribution specifies the probability of sampling each pair.

Note that the LWE distribution is not the same as the product measure in the probability space $A_{s,\chi}$ is defined on.

Definition 25 (search-LWE problem)**.** Let $s \leftarrow \mathbb{Z}_q^n$ be random. Let $(\mathbf{A}, b) \leftarrow \text{LWE}_{s,n,q,\chi,m}$. The (average-case) search-LWE problem is to construct a PPT algorithm A such that

$$\Pr[A(\mathbf{A}, b) = s] \neq \text{negl}(n) \tag{1.1}$$

Definition 26 (decision-LWE problem)**.** Let $s \leftarrow \mathbb{Z}_q^n$ be random. Let $\mathcal{U}_n$ be the uniform distribution on $Z_q^{n\times m} \times Z_q^m$. The (average-case) decision-LWE problem is to distinguish $\mathcal{U}_n$ and $\text{LWE}_{s,n,q,\chi,m}$ with non negligible advantage w.r.t n Connect to ensembles being computationally indistinguishable

Remark. Since the requirement is that s is randomly chosen instead of arbitrarily chosen, the problem refers to average-case hardness as opposed to worst-case hardness. Although this appears to be careless in the context of cryptography, it is actually equivalent. In other words, if average keys are easy to guess, then all keys are easy to guess. See [Reg10] for proof.

The LWE search and decision problem are equivalent when q is prime and $q(n) = \text{poly}(n)$.

The decision version of Learning With Errors (LWE) problem is based on distinguishing the uniform ensemble from the the LWE ensemble with distributions

1.5 The four generations of HE

Gentry did good rundown. See https://www.youtube.com/watch?v=487AjvFW1lkt=2s

If you are using mendeley to manage references, you might have to export them manually in the end as the automatic ways removes the ”date accessed” field

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.

[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.

[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.

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

[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.

[Hal17] Halevi, Shai. “Homomorphic Encryption”. In: *Tutorials on the Foundations of Cryptography*. Ed. by Yehuda Lindell. Information Security and Cryptography. Cham: Springer International Publishing, 2017. Chap. 5, pp. 219–268. ISBN: 978-3-319-57047-1. DOI: 10.1007/978-3-319-57048-8. URL: https://doi.org/10.1007/978-3-319-57048-8.

[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.

[LNP22] Li, Yang, Ng, Kee Siong, and Purcell, Michael. *A Tutorial Introduction to Lattice-based Cryptography and Homomorphic Encryption*. 2022. arXiv: 2208.08125 [cs.CR].

[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.

[Reg10] Regev, Oded. “The learning with errors problem (invited survey)”. In: *IEEE Conference on Computational Complexity*. 2010, pp. 191–204.

[Shr04] Shreve, Steven E. *Stochastic Calculus for Finance II: Continuous-Time Models*. Springer, 2004. ISBN: 0-387-40101-6.

[Vai11] Vaikuntanathan, Vinod. “Computing Blindfolded: New Developments in Fully Homomorphic Encryption”. In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*. Supported by an NSERC Discovery Grant and by DARPA under Agreement number FA875011-2-0225. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. University of Toronto. IEEE, 2011. DOI: 10.1109/FOCS.2011.98.

Contents

Appendix A

First Appendix

This is only slightly related to the rest of the report

Appendix B

Second Appendix

this is the information