Fall 11-7-2013

FULLY HOMOMORPHIC ENCRYPTION FOR WIRELESS NETWORK

Olive Mbianda
Southern Illinois University Carbondale, olivembianda.24@siu.edu

FULLY HOMOMORPHIC ENCRYPTION

FOR WIRELESS NETWORK

by

Olive Mbianda

B.S., NASPT- Yaounde, 2011

A Research paper
Submitted in Partial Fulfillment of the Requirements for the
Master of Science Degree

Department of Mathematics
in the Graduate School
Southern Illinois University Carbondale
December 2013

RESEARCH PAPER APPROVAL

FULLY HOMOMORPHIC ENCRYPTION FOR WIRELESS NETWORK

By

Olive Mbianda

A Research paper Submitted in Partial

Fulfillment of the Requirements

for the Degree of

Master of Science

in the field of Mathematics and Computer Science

Approved by:

Dr Kathy Spector, Chair

Dr Gregory Budzban

Dr John McSorley

Graduate School
Southern Illinois University Carbondale
November 07, 2013

AN ABSTRACT OF THE RESEARCH PAPER OF

OLIVE MBIANDA, for the Master of Science in MATHEMATICS AND COMPUTER SCIENCE, presented on NOVEMBER 07,2013 at Southern Illinois University Carbondale.

TITLE: FULLY HOMOMORPHIC ENCRYPTION APPLIED TO WIRELESS NETWORK

MAJOR PROFESSOR: Dr. K. SPECTOR, Dr. K. AKKAYA .

This work provides a mathematical approach of the Fully homomorphic encryption (FHE) and its implementation in a wireless network. FHE has been presented as the "Holy Grail" by the cryptographers. This special encryption scheme enables one to perform complex operations(both addition and multiplication) on a cypher text without ever decrypting the text. An immediate application is the delegated computation, an untrusted party can process the data without endangering the privacy of the source and the integrity of the data. The first FHE scheme was introduced in 2009, by Craig Gentry. His scheme was based on the properties of rings especially on ideal lattices.As introduced by Gentry, FHE was not practical due to the length of ciphertext (per bit encrypted) and the keys, and its infeasible computational time. Many works have been done to make it somewhat practical(Shai-Halevi(2010), Smart-Vercauteren(2011)).The proposed schemes were based on algebra and number theory concepts. Following the idea of Smart-Vercauteren, and the implementation of Michael Brenner we design an implementation for wireless network. Such a system should allow operations on encrypted data that could result in reducing the computation load and the size of the packets in a wireless network.The most challenging part of this work will be to make the computational time of the FHE quasi real while preserving its security scheme. Since the strength of the FHE comes from the hardness to

approximate short vector problems on arbitrary lattices within a slightly super polynomial factor, making that computational time logarithmic or less is quite challenging. This work attempts to design and implement fully homomorphic encryption for wireless networks.

ACKNOWLEDGMENTS

I would like to thank Dr. Tall for his invaluable assistance and insights leading to the writing of this paper. My sincere thanks also goes to Dr. Spector and Dr. Akkaya and the members of my graduate committee for their patience and understanding during the two years of effort that went into the production of this paper.

A special thanks also to Prof. Michael Brenner, from whose papers the java code used in this work have been derived. Another special thanks to Dr. Henry Hexmoor who has been very helpful and supportive.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Encryption is an "efficient" and well-known way for preserving the privacy of sensitive information sent through a network. The necessity for an encryption scheme allowing total privacy of data has become of greatest interest among cryptographers over the last few decades, due to recent development in the area of computer and mobile network. People were seeking for an encryption scheme that could allow operations on a ciphertext without any need to decrypting it first. In 1977, Rivest, Adleman and Shamir proposed a scheme (RSA) in which given only the public key and the encryption of operands, one could compute the encryption of their products. Therefore, the question of an encryption scheme allowing both addition and multiplication on the ciphertext arose. More concretely, was it possible to process encrypted data that is query it, write into it, and do any sort of operations that can be expressed as a circuit? Such a scheme known as Fully Homomorphic Encryption (FHE)today, was introduced in 1978 by Rivest, Adleman and Dertouzous. In their paper, they considered a situation in which a loan company enlists the services of a third-company to store and process its records. The loan company's database contains sensitive data and must be encrypted to ensure their privacy; so the third-party company is storing encrypted data. Now, let assume the loan company would like to know how much the average loan was or how many loans over 200 dollars were granted, but they don't have enough resources to compute such operations and need the help of the third-company to process it. How could the company get those statistics without endangering the privacy of the bank's users? Delegating computations to an untrusted party, that is allowing it to carry out extensive computation only on encrypted data, was the main goal of FHE. Nowadays the range of its applications has increased. Many attempts to produce such an encryption scheme have been made, but the real breakthrough came with Gentry in 2009. He proposed the first FHE scheme (Gentry, 2009) using ideal lattices. Ideal lattices correspond to ideals

in polynomial rings and they inherit natural additions and multiplications from the ring. Gentry suggested a public-key encryption scheme where the public and private key were respectively "bad" and "good" bases of an ideal lattice, and a small noise component was added to the text to be encrypted. The main issues with Gentry's scheme were the relatively extended length of the generated ciphertext and the large size of the encryption/decryption keys, both leading to an infeasible computational time. The practicability of FHE was questioned aroused (Fan & Vercatauren, 2010) since computing homomorphically caused the noise to increase leading to the failure of decryption. Since 2009, much research has been conducted to make FHE practical. An attempt to solve the problem was to make use of a somewhat homomorphic encryption leaving out the bootstrapping step, a partially decryption of the ciphertext to reduce the noise like in Gentry's scheme (Lauter et al., 2012). But, this turns out to be suitable only for a limited number of applications like private health care and online ads. In March 2012, a new somewhat homomorphic encryption scheme was proposed by Yang and Xia. It reduced the key size from O(k7) to O(k3) based on the approximate GCD problem, making it practical for cloud computing. In May 2012, an efficient fully encryption scheme leading to a public key size of O(k) was proposed by Brakersy and Vaikuntanathan. This encryption scheme is based on the learning with error assumption (Brakersi et.al, 2012). In June 2012, Michael Brenner et al. implemented a version of FHE based on Smart-Vercauteren approach (Brenner et al., 2012). The practical FHE has led to a large number of applications in computer networks: secure multi party computation (Kamara et al., 2012), private information retrieval (Dschai & Parski, 2010), delegated computation (Chung & Kalai, 2010). In our work, we are going to follow the implementation of Michael Brenner et.al, and use FHE for mobile wireless network, more specifically for smart phones.

We would like to apply FHE to mobile crowd sensing with smart phones. Mobile phone sensing is a new paradigm growing with the development of smart phones. Data are collected from the users and are processed by an external party for different purposes namely

traffic and weather monitoring. The challenge in this system is to preserve the privacy of the source. Though,this could be done by encryption, still the integrity of the collected data should be ensured. This means that after the encryption/decryption process, the result should be as similar as possible to the original data. Solutions have been proposed to address this issue ranging from PiRi, a privacy-aware framework for Participatory Sensing systems (Kazemi & Shahabi, 2011) to protocols (Moffat et al., 2011). FHE has been left out as a potential solution because of its complexity, but with its recent improvements we would like to investigate if a suitable version of FHE designed specifically for mobile networks can be derived from an existing practical FHE. Since many smart phone users are reluctant to participate in crowd sensing because of the privacy issue, our aim in this study is to provide the crowd sensing area with an efficient technique of users' privacy and data trustworthiness for the users.

Chapter 1 deals with cryptosystems settings. We will discuss security-related issues of encryption schemes and provide examples of private and public cryptosystems as well as their underlying security assumptions. This chapter also contains materials on groups, integers and functions. The description of homomorphic encryption is provided as well and is illustrated by two examples of additive and multiplicative homomorphic encryptions.

Chapter 2 deals with FHE. We will present the mathematical foundation of FHE: rings, fields and lattices. Then we will discuss some problems classified as hard in Number-theory and Algebra and being used as security assumptions for FHE encryption schemes. We will finally present three different FHEs based on lattices, integers, and learning with error.

Chapter 3 introduces our suggested algorithm. It is a combination of the three algorithms above-mentioned. We will give a complete description of the scheme, generation of the keys, encryption and decryption functions, verification of fully homomorphic properties as well as security assumptions.

CHAPTER 2

BACKGROUND

2.1 BACKGROUND IN CRYPTOGRAPHY

This section will provide some basic knowledge about cryptography. We will explain what is cryptography, why it is useful and provide some related vocabulary.

Alice and Bob are two friends and they would like to exchange messages over an insecure channel. The channel is considered insecure whenever it is feasible for Trudy (an adversary) to have access to the conversation. Since, they are aware of a potential eavesdropper, Trudy, trying to break into their conversation, they decide to encrypt it. The goal of encryption is to keep information secret from all, except to the authorized users. They might decide to map every bit or set of bits of their conversation to another bit or set of bits. A simple example would be the Caesar cipher. This encryption scheme is attributed to the emperor Caesar; He used it to secretly transmit his strategy to his troops in the field. In the Caesar cipher, each letter of a message is replaced with another letter of a fixed number of places after it in the alphabet. **Example** Bob wants to send the word **ATTACK** to Alice, he could chose to replace each letter by the third letter after it , so that A will be replaced by D,etc... so that ATTACK will be send as **DWWDFN**. Alice would know she will have to go back three letters to recover the original message.

We will call the original message(the one sent by Bob) **a plaintext**, and the encrypted version received by Alice **a ciphertext**, and the shift by 3 the **key**. **Cryptography** is the art and science of designing secure cryptosystems to guarantee "secure" communication over an insecure network. This mean,they should be guaranteed that after encrypting the data, they could always and are the only one(s)to decrypt it (i.e to reverse the mapping.).

A cryptosystem is a quadruple S=(M,C,K,$\mathcal{E}, \mathcal{D}$) such that :

1- M,C, and K are sets, where M is the message space (or "plaintext" space), C is the ciphertext space and K is the key space.

2- $\mathcal{E} = \{E_k | k \in K\}$ is a family of functions $E_k : M \to C$ that are used for encryption and $\mathcal{D} = \{D_k | k \in K\}$ is a family of functions $D_k : C \to M$ that are used for decryption.

3- For each key e $\in$ K, there exists a key d $\in K$ such that for each message m $\in m$: $D_d(E_e(m)) = m$, where **e** and **d** are respectively called the encryption and decryption key.

Let consider our previous example: Alice and Bob, communicating over an insecure channel and therefore using a cryptosystem S=(M,C,K,$\mathcal{E}, \mathcal{D}$) as defined above. Let assume the messages are distributed on M according to a probability distribution Pr_m(that may depend on the language used). For each new message m, Alice chooses a new key from K that is independent of the message to be encrypted (the key is usually generated before the plaintext). The keys are distributed according to a probability distribution Pr_k on K. The distributions Pr_m and Pr_k induce a probability distribution: $Pr_{M\times K}$ on $M \times K$. That is for each message $m \in M$ and for each $k \in K$,

$$Pr_{M\times K} = Pr_M(m)Pr_K(k)$$

is the probability that the message m is encrypted with the key k, where m and k are independent. Then we have:

- For $m \in M$, let m denote the event $\{(m, k) | k \in K\}$. Then $Pr(m) = Pr_M(m)$ is the probability that the message m will be encrypted.
- For $k \in K$, let k denote the event $\{(m, k) | m \in M\}$. Then $Pr(k) = Pr_K(k)$ is the probability that the key k will be used.
- For $c \in C$, let c denote the event $\{(m, k) | E_k(m) = c\}$.
  Then, $Pr(m|c)$ is the probability that m is encrypted given that c is received.

Kerchoff's principle: The security of a cryptosystem must not depend on the secrecy of the system used. Rather, the security of a cryptosystem may depend only on the secrecy of the keys used.

Definition 2.1.0.1. (**Perfect secrecy**)

A cryptosystem

$S = (M, C, K, \mathcal{E}, \mathcal{D})$ is said to guarantee **perfect secrecy**,

$$\text{iff} \quad \forall m \in M, \forall c \in C, Pr(m|c) = Pr(m)$$

The **perfect secrecy** is the aim of any cryptosystem.

Theorem (Shannon)

Let S=(M,C,K,$\mathcal{E}, \mathcal{D}$) be a cryptosystem with $|C| = |K|$ and $Pr(m) > 0$ for each $m \in M$. Then S guarantees perfect secrecy iff:

1-For each $m \in M$ and for each $c \in C$, there exists a unique key $k \in K$ with $E_k(m) = c$ and,

2- The keys in K are uniformly distributed.

According to the Kerchoff's principle, in cryptography settings, provable inefficiency means security: The security of current cryptosystems usually depends on the assumption that certain problems from algebra and number theory are intractable. Thus, to describe such cryptosystems and discuss related security issues, we need algebraic and number-theoretical notions, and results. In the next section we will give some background on groups, integers and present some related hard problems used in cryptography.

2.2 MATHEMATICAL BACKGROUND

This section will provide mathematical notions and useful theorems necessary to clearly understand fully homomorphic encryption algorithms and their security assumptions. We will start by giving some algebraic and number theory foundation, then we will discuss integers and theorems around them.

2.2.1 Background in Algebra

Groups

Definition 2.2.1.1. *A group* G is a non-empty set (G, $*$) together with a binary operation G$\times G \to G$.

$(\alpha, \beta) \to \alpha * \beta$, (closure of the group)such that the following holds:

a) **Associativity**

$\forall \alpha, \beta, \gamma, \in G : \alpha * (\beta * \gamma) = (\alpha * \beta) * \gamma$

b) **Existence of an identity element e**

$\exists e \in$ G such that $\forall \alpha \in G : \alpha * e = e * \alpha = \alpha$. The identity element is unique.

c) **Existence of an inverse element**

$\forall \alpha \in G, \exists \alpha^{-1} \in G, \text{such that} \alpha * \alpha^{-1} = \alpha^{-1} * \alpha = e$

Example:

- Let define the group of plaintext (M, $\oplus$) where $\oplus$ is the XOR(Exclusive OR)or the addition mod 2 and M=$\{(0,1)^k\}$ where k is a positive integer.

  Let recall the logical table for XOR

$\oplus$	0	1
0	0	1
1	1	0

  Claim M is a group under $\oplus$, and we will call it he group of plaintexts under addition.

  Proof:

  * Associativity

  Let $m_1, m_2, m_3 \in \mathbb{Z}_2$, we have:

m_1	m_2	m_3	$m_1 \oplus (m_2 \oplus m_3)$	$(m_1 \oplus m_2) \oplus m_3$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Notice that $m_1 \oplus (m_2 \oplus m_3) = (m_1 \oplus m_2) \oplus m_3$, thus the associativity is satisfied.

* Identity element is 0

Notice that $0 \oplus 0 = 0$ and $1 \oplus 0 = 1$.

* Inverse element

$0 \oplus 0 = 0$ so that 0 is the inverse of 0 and $1 \oplus 1 = 0$ so that 1 is the inverse of 1, every element of M has an inverse in M. We can conclude that **(M,⊕) is a group**.

- Is $(\mathbb{Z}_2, \cdot)$ a group, where $\cdot$ is the multiplication mod 2?

Let recall

$\cdot$	0	1
0	0	0
1	0	0

We notice that we have an identity element **1**, but not all element has an inverse for instance **0**, thus $(\mathbb{Z}_2, \cdot)$ is not a group

Definition 2.2.1.2. *A semi group* is an algebraic structure consisting of a set together with an associative binary operation

$(\mathbb{Z}_2, \cdot)$ is a **semi-group**, and since it has an identity element it is referred to as **monoid**. The requirement for cryptography settings is the closure of the groups, associativity and the existence of the identity element, so we will be working with monoids and groups.

2.2.2 Functions

Let recall that a binary relation from a non-empty set A to a set B is any subset of A × B, where A×$B = \{(a, b)|a \in A$ and $b \in B\}$.

Definition 2.2.2.1. *A function f* is a binary relation between a set A(set of inputs) and a set B (valid outputs). Each input has exactly one output. In other words if (a,b_1) ∈ f and (a,b_2) ∈ f, then $b_1 = b_2$.

Let M, C, K be respectively the sets of plaintexts, ciphertexts, and keys then we define:

$$\textbf{Encrypt:=} \quad E_K : M \times K \to C$$
$$(m, e) \mapsto c,$$

$$\textbf{Decrypt:=} \quad D_K : C \times K \to M$$
$$(c, d) \mapsto m,$$

The functions Encrypt and Decrypt should both be easy to compute, and moreover should guarantee the perfect secrecy that is we must have ∀ m ∈ M, d,d' ∈ K with $d \neq d'$ **Decrypt(Encrypt(m,e),d')≠ m**

2.2.3 Homomorphisms

Definition 2.2.3.1. A homomorphism is a mapping ϕ between two groups (G,♢) and (H, ∗) such that ϕ (x ♢ y)=ϕ(x) ∗ ϕ(y) for x,y ∈ G and ϕ(x),ϕ(y) ∈ H. Such a function ϕ is called *a homomorphic function.*

Example: Let define the function $\phi : M \rightarrow$ C, and $\phi(m) = m^e$ where e is an integer. We easily verify that $\phi(m_1 \cdot m_2) = (m_1 \cdot m_2)^e = {m_1}^e \cdot {m_2}^e = \phi(m_1) \cdot \phi(m_2)$

Definition 2.2.3.2. A multiplicative(resp. additive) homomorphic function is a homomorphic function with respect to multiplication (resp. addition), i.e $\phi(x \diamondsuit y) = \phi(x) \cdot \phi(y)$ (resp. $\phi(x \diamondsuit y) = \phi(x) + \phi(y)$), for $x, y \in G$ and $\phi(x), \phi(y) \in H$.

Following definition 1.2.3.2,

An encryption scheme would be said to be *additively homomorphic* if the following holds:

(i) Decrypt $(c_1 +_C c_2)$**= Decrypt**(c_1) $+_P$ **Decrypt**(c_2) for ciphertexts c_1 and c_2

(ii) Encrypt(m_1) $+_C$ **Encrypt**(m_2) "is like" Encrypt$(m_1 +_P m_2)$ where m_1 and m_2 are two plaintexts.

Similarly, an encryption scheme would be said to be **multiplicatively homomorphic** if we have:

(i) Decrypt $(c_1 \times_C c_2)$**= Decrypt**(c_1) $\times_P$ **Decrypt**(c_2) for ciphertexts c_1 and c_2

(ii) Encrypt(m_1) $\times_C$ **Encrypt**(m_2) "is like" Encrypt$(m_1 \times_P m_2)$ where m_1 and m_2 are two plaintexts.

Earlier homomorphic schemes were homomorphic with respect to either addition or multiplication but not to both at the same time. Before going into details of homomorphic encryptions, it is necessary to give some background on integers, since most cryptographic security assumptions rely on hardness of some mathematical problems related to integers. The next section provides some useful results and theorems which are essential for cryptosystems.

2.2.4 Background on integers

Most encryption schemes rely on the properties of integers to provide encryption and decryption algorithms as well as ensuring the security of the scheme. It is important to present some useful properties of integers.

Definition 2.2.4.1. Let S be a set , a relation R, a and b $\in$ S. R is an equivalent relation on S if the following properties hold:

(i) Reflexivity aRa

(ii) Symmetry If aRb $\Rightarrow$ bRa

(iii)Transitivity If aRb and bRc $\Rightarrow$ aRc

We usually denote R by $\sim$

The equivalence class of a under $\sim$, denoted by [a] is defined as [a]=$\{b \in A | a \sim b\}$

Divisibility

For every two integers a and b with b$\neq$ 0 a =r+qb with $r < b$ where r is the remainder of the division of a by b.

Definition 2.2.4.2. We say b divides a if r=0, i.e a=qb, b is called a **divisor** of a; and a is called a **multiple** of b. We denote by **[0]** the class of integers for which the remainder of the division by n is 0 and by **[1]** the class of integers for which the remainder of the division by n is 1. These equivalence classes are called residue classes modulo n.

Definition 2.2.4.3. "a is congruent to b modulo n" denoted **a**$\equiv b \bmod n$ iff a-b is divisible by n.

Let show that **congruence** is an equivalence relation.

1- **Reflexive property :** a$\equiv a \bmod n$ since a-a=0 is divisible by n.

2- **Symmetric property:** if a$\equiv b \bmod n$, then b$\equiv a \bmod n$, since if a-b is divisible by n then b-a=-(a-b) is divisible by n.

3-**Transitive property:** if a$\equiv b \bmod n$ and b$\equiv c \bmod n$, then a$\equiv c \bmod n$ since $\frac{a-b}{n} = q_1$, where q_1 is an integer, and also $\frac{b-c}{n} = q_2$, where q_2 is an integer, then $\frac{a-b}{n} + \frac{b-c}{n} = q_1 + q_2 = \frac{a-c}{n} \Rightarrow \frac{a-c}{n} = q_1 + q_2$ where $q_1 + q_2$ is an integer

We denote by $\mathbb{Z}_n$=$\{$[0],[1],[2],...[n-1]$\}$ and by $\mathbb{Z}_{n^2}$=$\{$[0],[1],[2],...$[n^2 - 1]\}$.

Definition 2.2.4.4. Let a $\in \mathbb{Z}_n$, a has an **inverse** in $\mathbb{Z}_n$ if $\exists b \in \mathbb{Z}_n$ such that $a \cdot b = 1 \bmod n$.

We denote by $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n / \exists b \in \mathbb{Z}_n \ a \cdot b = 1 \bmod n\}$.

Definition 2.2.4.5. The greatest common divisor (gcd) of a and b , denoted by **gcd(a,b)** or simply **(a,b)** is a positive number **d** such that d/a and d/b and if x is any integer such that x/a and x/b then x/d .

The gcd (a,b) always exists and is unique.

Theorem: (Bezout's theorem) Let a and b two nonnegative integers, a and b are relatively prime if $\exists$ u, v $\in \mathbb{Z}$ such that au+bv=1.

Proof

Let S=$\{au + bv > 0$ a,b $\in \mathbb{N}\}$. There exists a least element d $\in$ S such that au+bv=d.

* S $\subseteq \mathbb{N} \Rightarrow S \neq O$

* We want to show that d=1

$d = au_1 + bv_1$ (1), let consider the Euclidean division of a by d, a= dq + r (2) with $0 \leq r < d$ (3).

(1) in (2)$\Rightarrow$ a= $(au_1 + bv_1)q + r$

$\Rightarrow$ a - au_1q - bv_1q= a(1-u_1q)- b(v_1q)=r $\Rightarrow$ r=au_1' + bv_1' $\Rightarrow$ r $\in$ S and $r < d$ (from (3)) which is a contradiction, so r must be 0, thus d divides a.

We apply a similar reasoning for b, so d divides b and d divides both a and b.

It follows $\frac{d}{d} = \frac{a}{d}u_1 + \frac{b}{d}v_1 \Rightarrow$ 1= a'u_1 +b'v_1 (a' and b' $\in \mathbb{Z}$ since d divides a and d divides b).

1= a'u_1 +b'v_1 $\Rightarrow$ 1$\in$ S and 1 $\geq$ d (since d is the least element of S), so finally we have d=1.

Definition 2.2.4.6. The least common multiple (lcm) of a and b, denoted by **lcm(a,b)** is a positive number **l** such that a/l and b/l and if a/x and b/x then l/x.

Note: The lcm of two positive integers always exists and is unique.

Definition 2.2.4.7. A **prime number** is a positive integer greater than 1 that has no positive divisors other than 1 and itself.

Definition 2.2.4.8. a and b are said to be **coprime** or **relatively prime** if their only common divisor is 1.

Definition 2.2.4.9. *The Euler totient function* $\phi(n)$ is the number of integers less than or equal to n and which are relatively prime to n.

Example: Let calculate $\phi(6)$, 1,2,3,4,5 are integers less than 6 but only 1 and 5 are relatively prime to 6, thus $\phi(6) = 2$

Theorem : Let $\phi(n)$ be the Euler totient function and n be an integer. Then $\phi(n)$ has the following properties:

1- ϕ is a multiplicative function : if m and n are relatively prime then $\phi(mn) = \phi(m)\phi(n)$

2- For p prime and k ≥ 1 where k is an integer : $\phi(p^k) = (p-1)p^{k-1}$.

3- $\phi(n^k) = n^{k-1}\phi(n)$ **Theorem: Fermat's little theorem**

For any prime p, and any integer $a \neq 0 \pmod{p}$, we have $a^{p-1} \equiv 1 \bmod p$

Definition 2.2.4.10. *The Carmichael's function* For a positive integer n, $\lambda(n)$ denotes the least positive integer t such that $m^t \equiv 1 \bmod n$ for all integers m with gcd(m,n)=1. $\lambda(n)$ as defined above is called the Carmichael function.

Example Let compute $\lambda(6)$. We have gcd(1,6)=gcd(5,6)=1, then we have $1^n \equiv 1 \bmod 6$ for $n \geq 1$, and $5^2 \equiv 1 \bmod 6$, and since we are looking for the least integer, we will set n=2 and thus $\lambda(6) = 2$

This section provides very important notions to understand the mathematical foundation of Paillier cryptosystem.

2.2.5 Integer factorization problem

Definition 2.2.5.1. *A composite number n* is a positive integer $n > 1$ such that n is not prime i.e n can be divided evenly by other numbers (other than 1 and itself).

Definition 2.2.5.2. Let N be a composite integer. There exists integers u,v such that

N=$u \cdot v$ and such that both $u, v > 1$.

u and v are called **factors**.

Definition 2.2.5.3. *Computational Integer factorization.* Given an integer N and an integer M with $1 \leq M \leq N$, does N have a factor d with $1 < d < M$?

When the numbers are very large, no efficient integer factorization algorithm is publicly known. Not all numbers of a given length are equally hard to factor. The hardest instances of these problems are semi-primes, the product of two prime numbers, when they are both large, randomly chosen and about the same size.

In the following section we will consider n=pq where p and q are prime numbers, then $\phi(n) = \phi(pq) = \phi(p)\phi(q)$ and $\lambda(n) = lcm(p-1, q-1)$ where $\phi(n)$ and $\lambda(n)$ are respectively the Euler totient function and the Carmichael's function.

The composite residue problem

Let recall that $\mathbb{Z}^*{}_{n^2}$={ a$\in \mathbb{Z}_{n^2}/\exists$ b$\in \mathbb{Z}_{n^2}$, a$\cdot$ b $\equiv 1 \bmod n^2$}.

Let g be some element of $\mathbb{Z}^*{}_{n^2}$ and denote by ε_g the integer valued function defined by:

$$\begin{array}{rl} & \mathbb{Z}_n \times \mathbb{Z}_n^* \to \mathbb{Z}^*{}_{n^2} \\ \varepsilon_{g(x,y)} & \\ & (x, y) \mapsto g^x \cdot y^n \bmod n^2 \end{array}$$

We denote by $\mathcal{B}_\alpha \subset \mathbb{Z}^*{}_{n^2}$ the set of elements of order nα and by $\mathcal{B}$ their disjoint union

Definition 2.2.5.4. A number z is said to be **a n-th residue modulo** n^2 if there exists a number y $\in \mathbb{Z}_n^*$ such that z $\equiv y^n \pmod{n^2}$.

Definition 2.2.5.5. Assume that g $\in \mathcal{B}$. For w $\in \mathbb{Z}^*{}_{n^2}$, **n-th residuosity class of w** with respect to g the unique integer x $\in \mathbb{Z}^*_n$ such that $\varepsilon_g(x, y) = w$.

Definition 2.2.5.6. A composite residuosity class problem is the computational class problem defined as follows: given w$\in \mathbb{Z}^*{}_{n^2}$ and g $\in \mathcal{B}$, compute $[[w]]_g$, where $[[w]]_g = \dfrac{L(w^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2))} = \dfrac{w^\lambda \bmod (n^2) - 1}{g^\lambda \bmod (n^2) - 1}$.

2.3 MORE ON CRYPTOGRAPHY

Alice and Bob (from the section 1) wants their conversation to remain secret from the eavesdropper Trudy, so they mapped the real conversation into another one. This mapping protects them from any intruder, now the mapping has to be easier to reverse for Alice (assuming that Bob is the sender). There is one big issue associated to the mapping, the way of reversing it, has to be known by each protagonists but remain unknown by Alice and Bob. From our previous example(see section 1), how Bob would tell Alice to replace the letter by the one in the third position above it?

An immediate solution would be for Alice and Bob to change their keys in person and use the same key for all their conversations. In the reality, there might be no possibility for Alice and Bob to communicate in person (that's the main reason why we assume they communicate through a network), so practically how could they agree on the keys? Would they use the same key for encryption and decryption?

2.3.1 Symmetric-key cryptography

Definition

An encryption scheme is said to be **symmetric** if **a secret key k** $\in$ K is shared by the sender and the receiver, i.e k=e=d

The Caesar cipher, mentioned in section 1 is an example of a symmetric-key encryption. In our example the shared key is 3, and the encryption function is **c=m+3**, while the decryption function is **m=c−3**

Example of a symmetric-key encryption scheme

AES

2.3.2 Public-key or asymmetric key cryptography

Definition

An encryption scheme is said to be asymmetric if **a public key** e $\in K$ used by the sender to encrypt the message and a **secret key** d $\in$ K used by the receiver to decrypt the message, i.e $e \neq d$.

An analogy to the asymmetric cryptography is that everyone can send a letter to Alice (using her mailing address, it is publicly known), but only Alice can read the letter (she is the only one to have the key of her mailbox).

We will present two examples of public-key encryptions which happen to be homomorphic also.

2.3.3 RSA

Rivest **S**hamir and **A**dleman (RSA) is a public encryption created in 1977 by Ron Rivest, Adi Shamir and Len Adleman at MIT. The security of the scheme is based on the difficulty of factoring large integers. By the fundamental theorem of arithmetic, every positive integer has a unique prime factorization. The most difficult integers to factor in practice are those that are products of two large primes of close size. Another interesting property of RSA is its multiplicative homomorphism.

Let recall that a scheme is multiplicative homomorphic if for a plaintext m=$m_1 \times_M m_2$ **Decrypt(Encrypt(m))**=Decrypt(Encrypt($m_1 \times_P m_2$))=Decrypt($c_1 \times_C c_2$)= $m_1 \times_M m_2$.

In simpler words, that is given only the public key and the encryption of m_1 and m_2, one can compute the encryption of $m_1 \times_M m_2$

Description of the RSA cryptosystem

We will assume Bob and Alice are communicating, and Alice is sending a message to Bob.

Step 1: KeyGen (Generation of the keys)

- **Input:** Bob chooses two large distinct prime numbers p, q

Then he computes n=pq and $\phi(n) = (p-1)(q-1)$.

He chooses **e** such that $1 < e < \phi(n)$, gcd(e,$\phi(n)$) = 1

Finally he determines the inverse element of e mod $\phi(n)$, i.e the unique number **d** such that $1 < d <$ (mod ϕ)(n) and $e \cdot d = 1 \pmod{\phi}(n)$

e and n are called respectively the encryption and the decryption exponents

- **Output:** public and private(secret) keys (pk,sk)

Bob's public key is pk=(e,n), and his secret key is sk=d

Now, Alice knows Bob,s public key, she will use that to encrypt a message and send it to him.

Step 2: Encrypt (m, pk)

- **Input:** plaintext m $\in \mathbb{Z}_n$ and public key pk=(e,n)

Alice wants to send m$\in \mathbb{Z}_n$

She computes $c = E_{(n,e)}(m)$ where $E_{(n,e)}(m) = m^e \pmod{n} \forall m \in \mathbb{Z}_n$

- **Output:** ciphertext c $\in \mathbb{Z}_n$

Now Alice sends c to Bob, Bob will receive c and will apply the decryption function to recover m.

Step 3: Decrypt(c,sk)

- **Input:** ciphertext c $\in \mathbb{Z}_n$ and secret key sk =d

He computes $D_d(c) = c^d \pmod{n}$

$$\begin{aligned} m^{ed} &= m^{ed} \\ &= m^{1+k(p-1)(q-1)} \\ &= m(m^{p-1})^{k(q-1)} \end{aligned}$$

Then $m^{ed} = m \pmod{p}$ and if p does not divide m, then we have $m^{p-1} = 1 \pmod{p}(1)$ (by Fermat little theorem), by a symmetric argument we show that $m^{ed} = m \pmod{q}$ (2). Since p and q are distinct primes, from (1) and (2)we have $m^{ed} = m \pmod{p}$

- **Output:** m $\in \mathbb{Z}_n$

Multiplicative homomorphic property of RSA

Claim: RSA is multiplicatively homomorphic i.e Enc(m_1) $\times_c$ Enc(m_2)=Enc($m_1 \times_p m_2$)

Proof Let consider the groups (M,$\cdot$) and (C,$\cdot$) to be respectively the groups of plaintext, and ciphertext. Let recall the encryption function:

$$(M, \cdot) \to (C, \cdot)$$

$$m \mapsto m^e \pmod{n}$$

Let m $\in$ M such that $m = m_1 \cdot m_2$ where $m_1, m_2 \in M$,

We have

$$Enc(m_1) \cdot Enc(m_2) = {m_1}^e \pmod{n} \cdot {m_2}^e \pmod{n}$$

$$= m_1 \cdot {m_2}^e \pmod{n}$$

$$= m^e \pmod{n}$$

$$= Enc(m)$$

Bob would like to send a message to Alice, for instance w (the ascii code is 119). He wants Alice to be the only person able to read the message.

1- He chooses two (large) distinct primes number p and q at random

p=29 and q=31

2- He computes the RSA modulus n=pq

n=31 $\times$ 29=899

3-He computes $\phi(n) = (p-1)(q-1)$

$\phi(n) = 28 \times 30 = 840$

4-He selects a random integer e such that gcd(e,$\phi(n)) = 1$

we choose e to be 11 since gcd(11,840)=1

5-He computes the unique integer d such that $ed \bmod \phi(n) = 1$

we have $11d \bmod 840 = 1 \Rightarrow d = 611$.

6- The public key is (e,n), the private key is (d,n)

(11,899) is the public key and **(611,840)** is the private key.

Step 2: Encrypt

- **Input:** plaintext p and public key pk

- **Output:**ciphertext c, $c = p^e \bmod n$

- **Function:** Encrypt (p, pk)

Bob sends to Alice c=$119^{11} \bmod 899 = 595$

Step 3: Decrypt

- **Input:** ciphertext and secret key sk

- **Output:** plaintext p, $p = c^d \bmod n$

- **Function:** Decrypt (c, sk)

Alice computes $p = 595^{611} \bmod 899 = 119$ and recovers the original message

Evaluate

Let consider a plaintext $m = m_1 \times m_2$. We have $c = m^e \bmod n = m_1 \times {m_2}^e \bmod n = {m_1}^e \times {m_2}^e \bmod n = {m_1}^e \bmod n \times {m_2}^e \bmod n = c_1 \times c_2$.

Let consider a ciphertext c which is such that $c = c_1 \times c_2, P = c_1 \times {c_2}^d \bmod n = ({c_1}^d \times {c_2}^d) \bmod n = {c_1}^d \bmod n \times {c_2}^d \bmod n = m_1 \times m_2$

Paillier cryptosystem

It was invented in 1999 by Pascal Paillier. It is a public-key crypto-system based on composite degree residue classes. The security of this scheme is ensured by the hardness of computing the n-th residue classes i.e given a composite n (n=pq where p and q are large prime numbers) and an integer z it is hard to decide whether z is a n-residue modulo n^2 or not, i.e whether $\exists$ y such that z $\equiv y^n \pmod{n^2}$. Moreover,**Paillier cryptosystem** is an **additive homomorphic** cryptosystem.

Let recall that a scheme is additive homomorphic if for a plaintext p=$p_1 +_P p_2$ we have **Encrypt(p)=Encrypt**$(p_1 +_P p_2)$=**Encrypt**(p_1) +**Encrypt**(p_2) and **Decrypt(Encrypt(p))**=$p_1 +_C p_2$. That is given only the public key and the encryption of m_1

and m_2, one can compute the encryption of $m_1 +_M m_2$. Notice that we don't need to know the original message, and such a scheme is useful if the cost to compute Encrypt(m_1) and Encrypt(m_2)is less than computing Encrypt(m_1)$+_C$ Encrypt(m_2).

In this section we will describe the Paillier's scheme and illustrate it by an example.

Description of the Paillier's scheme

The Paillier crypto system works as follows: Bob wants to send a message to Alice.

Step 1: KeyGen (Generation of the keys)

- **Input:** Two large prime numbers p, q $\in \mathbb{N}$

Compute n=pq

Choose g $\in \mathbb{Z}^*{}_{n^2}$ such that $gcd(L(g^\lambda \pmod{n^2}), n) = 1$ with L(u)=$\frac{u-1}{n}$

- **Output:** public and private(secret) keys (pk,sk)

pk=(n,g), sk=(p,q)

Step 2: Encrypt (m, pk)

- **Input:** plaintext m $\in \mathbb{Z}_n$ and pubblic key pk=(n,g)

Choose r$\in \mathbb{Z}^*{}_n$

Compute c= $g^m r^n \pmod{n^2}$

- **Output:** ciphertext c $\in \mathbb{Z}_{n^2}$

Step 3: Decrypt(c,sk)

- **Input:** ciphertext c $\in \mathbb{Z}_{n^2}$ and secret key sk =(p,q)

Compute m=$\frac{L(c^\lambda \pmod{n^2})}{L(g^\lambda \pmod{n^2})} \pmod{n}$

- **Output:** m $\in \mathbb{Z}_n^*$

Additively homomorphic property of the Paillier's cryptosystem

Claim: Paillier's cryptosystem is additively homomorphic i.e Enc(m_1) $+_c$ Enc(m_2)=Enc($m_1 +_p m_2$)

Proof Let consider the groups (M,+) and (C,$\cdot$) to be respectively the groups of plaintext, and ciphertext. Let recall the encryption function:

Enc(m) : (M,+) $\rightarrow$ (C,$\cdot$)

m $\mapsto$ c=$g^m r^n \bmod n^2$

Let m_1, m_2 be two plaintexts, then we have Enc(m_1) $\cdot$ Enc(m_2)=($g^{m_1} {r_1}^n \bmod n^2$) $\cdot(g^{m_2} {r_2}^n \bmod n^2)$ =$g^{m_1+m_2} r_1 {r_2}^n \bmod n^2$ since $r_1, r_2 \in \mathbb{Z}_n \rightarrow r_1 \cdot r_2 \in \mathbb{Z}_n$, let r=$r_1 r_2$, so we have Enc($m_1$) $\cdot$ Enc(m_2)=$g^{m_1+m_2} r^n \bmod n^2$=Enc($m_1 + m_2$), thus Paillier cryptosystem is an additively homomorphic.

Example: Bob wants to send a message to Alice. He will like to use the Paillier encryption scheme. In the following example we illustrate the steps.

1- **Bob chooses two large prime numbers randomly and independently of each other such that gcd(pq,(p-1)(q-1))=1**.

Bob chooses for instance p=31 q=17; then he computes n=pq=527 and t (p-1)(q-1)=480 and he checks that gcd (527, 480)=1

2- **Then he computes n=pq and λ=lcm(p-1,q-1)**

n= pq= 527 and λ= lcm(30,16)= 240

3- He selects a random integer where g $\in \mathbb{Z}^*_{n^2}$ and he makes sure that n divides the order of g, by checking if he can find $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ where the function L is defined as $\forall u \in \mathcal{S}_n$, L(u)=$\frac{u-1}{n}$ with S_n={ u $\leq n^2 \mid u \equiv 1 \bmod$ n}.

In other words we can set g=n+1, $\lambda = \phi(n) = (p-1)(q-1)$ and $\mu = \phi^{-1}(n) \bmod n$.

He selects g=1055

The public key pk is (n,g) so pk=(527,1055)

The private key sk is (p,q)=(31,17)

The first step is completed,we have computed the keys. Now let assume Bob wants to send m=50 , he chooses r=35 .

c=$1055^{50} 35^{527} \pmod{527^2}$=**165122**

Now let assume Bob doesn't have enough resources to compute m=50, so he writes m=$m_1 + m_2$=50 with m_1 =34, and m_2= 16. He also chooses r_1=5 and r_2=7.

He computes c_1 and c_2 and sends it to Alice. $c_1 = 1055^{34} 5^{527} \pmod{527^2}$=88220

$c_2 = 1055^{16}7^{527} \pmod{527^2}$=8760

Alice will receive c_1 and c_2 and will compute $c_1 \cdot c_2 = 88220 \times 8760 \pmod{527^2}$=**165122**. The second step is completed, Bob has encrypted the message and has send it to Alice.

Alice will use her secret key to decrypt the received message and recover the original message. she will compute: m=$\dfrac{L(165122^{240} \pmod{527^2})}{L(1055^{240} \pmod{527^2})} \pmod{527}$=**50**

2.4 TOWARDS FULLY HOMOMORPHIC ENCRYPTION

Earlier homomorphic encryption schemes were partially homomorphic, they were either additive or multiplicative.The idea of an encryption scheme allowing one to perform complex mathematical operations on a cipher text without ever decrypting the text was introduced by Rivest, Shamir and Adleman in 1978. Many attempts to produce such an encryption scheme have been made, but the real breakthrough came with Gentry in 2009. Fully homomorphic encryption is considered as the "Holy grail" by cryptographers due to the numerous applications it can lead to. Let take a practical example, there is an online software you can use to evaluate your insurance premium. The required inputs are your bank information, credit information, age, and some other sensitive information and you don't feel comfortable sending those information in clear through the network. Now, somebody assure you that, there is no need for you to send those data in clear, that you just have to encrypt your data(he has zero-knowledge of your data), and he will compute the function to evaluate your premium and would send you the encrypted result, and you again will be the only one to access your result. Among other things, a fully homomorphic encryption (FHE) scheme allows one to perform non-interactive secure computation, and in many applications this is crucial. The classic example is cloud computing: if you don't trust your cloud provider with your data, you are in trouble: either you have to give away your private data in clear (running the risk that the cloud provider looks into possibly confidential data), or you have to encrypt the data before uploading it (losing the

advantages of having the cloud computing for you). Another example is encrypted a Spam filter: you like that your mailbox is not filled with junk, but you might not be happy about Google/Microsoft/etc. reading the contents of all your email.

In the next chapter we will present some fully homomorphic encryption schemes along with their mathematical underlying structures and security assumptions.

CHAPTER 3

FULLY HOMOMORPHIC ENCRYPTION

3.1 EVOLUTION OF FULLY HOMOMORPHIC ENCRYPTION SCHEMES(FHE): FROM THE FIRST FHE TO PRACTICAL FHE

The need for more and more secure cryptosystems has increased drastically with technology, and in 1978, the idea of privacy homomorphism (today known as Fully Homomorphic Encryption (FHE)) was introduced by Rivest, Shamir and Dertouzous. That special encryption scheme should allow an unlimited chaining of algebraic operations, that means an arbitrary number of additions and multiplications can be applied to encrypted operands. The underlying question was to find if there was an encryption function such that both $Encrypt(x + y)$ and $Encrypt(x \cdot y)$ are easy to compute from $Encrypt(x)$ and $Encrypt(y)$? Finding such an encryption function was the promise of a whole bunch of applications,ranging from delegation of computation to untrusted parties to search on encrypted data. That is the reason why FHE is considered as the "HOLY GRAIL" in cryptography. Many attempts to produce such an encryption scheme have been made, but the real breakthrough came with Gentry. He proposed the first fully homomorphic encryption (FHE) scheme (Gentry, 2009),based on "ideal lattices". His scheme later on was not found practical, and researchers have been working to make FHE somewhat practical. The major concern in FHE is that, the noise introduced in the ciphertext to ensure security, increases with every single operation on the ciphertext lowering the accuracy of decryption and, eventually leading to its failure. To address this issue, cryptographers use "a hint" in the key to help refreshing the ciphertext, or simply use long ciphertexts (with unused bits). A somewhat homomorphic encryption (SHE) is a scheme in which no re-encryption is required(no need to refresh the ciphertext), but only a limited number of operations is

possible. An SHE is capable of evaluating "low degree" polynomials homomorphically. All known FHE encryption schemes are constructed from SHE. Thus, an FHE is obtained by squashing the decryption circuit, that is to use an encrypted secret key as a component of the public key and evaluate it under encryption by SHE. After squashing, the next step towards FHE is bootstrapping, which is partially decrypting ((refreshed) the ciphertext, and then use it in new homomorphic evaluations of low-degree polynomials. In this chapter, we will present the mathematics behind the FHE, we will provide some background on rings, fields, lattices then we will discuss two different FHE schemes based on lattices and integers.

3.1.1 Rings and Polynomials

Definition 3.1.1.1. A *ring* R is a non-empty set together with two operations + and · satisfying:

(i) The associative law for addition (a+b)+c=a+(b+c) $\forall a, b, c \in R$

(ii) The commutative law for addition a+b=b+a

(iii) The existence of 0: $\exists 0 \in R$ such that,$\forall a \in R a + 0 = 0 + a = a$

(iv) The existence of negatives: $\forall a, \in R \quad \exists -a \in R$ such that a+(-a)=0

(v) The associative law for multiplication (ab)c=a(bc) $\forall a, b, c \in R$

(vi) The distributive laws: a(b+c)=ab+ac, (a+b)c=ac+bc, $\forall a, b, c \in R$

A ring is said to be **commutative** if multiplication is commutative, i.e $a \cdot b = b \cdot a$, where a, b$\in R$

Example 3.1.1.1. $(\mathbb{Z}_2, +, \cdot)$ is a ring.

More generally $(\mathbb{Z}_p, +, \cdot)$ where + and · are respectively the addition and multiplication modulo p

Definition 3.1.1.2. Let $(R, +\cdot)$ be a ring with identity and let x be an indeterminate over

R. We denote by $R[x]$ the set of all formal expressions

$$a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x + a_0 \tag{2.1}$$

where n is a nonnegative integer and $a_j \in R$ for j=0,1,...n.

We define $1 \cdot x = x$ and for any $a \in R$, $a = ax^0$.

a_j is called the **coefficient** of x_j. Any expression of the form (2.1) is called a **polynomial** in x with coefficients in R.

Example 3.1.1.2. a) Let consider the ring ($\mathbb{Z}[x], +, \cdot$) we define $\mathbb{Z}[x]$ the set of polynomials with integer coefficients, $\mathbb{Z}[x] = \{f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n | a_i \in \mathbb{Z}$ with $0 \leq i \leq n\}$. a_n is called the **leading coefficient** and a_0 **the constant term**.

b) Similarly we define $\mathbb{Z}_p[x]$ to be the set of polynomials with coefficients in $\mathbb{Z}_p$ (p is a prime) in the ring ($\mathbb{Z}_p[x], +, \cdot$) , $\mathbb{Z}_p[x] = \{f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n | a_i \in \mathbb{Z}_p$ with $0 \leq i \leq n\}$.

We will be working with the ring $(Z_p[x], +, \cdot)$

Definition 3.1.1.3. For every nonzero polynomial, $p(x) \in R[x]$, there is a largest nonnegative integer $m(1 \leq m \leq n)$ such that $a_m \neq 0$ (recall that a_m is the coefficient of x^m), m is called **the degree of p(x)** and denoted **deg(p(x))**. a_m is called the **the leading coefficient** of p(x), a_0 is called **the constant term** of p(x)

A polynomial of the form $a_k x^k$ is called **a monomial**

A nonzero polynomial p(x) with deg(p(x))=n is called **a monic** if $a_n = 1$

Definition 3.1.1.4. A **division ring** is a ring in which every nonzero element has a multiplicative inverse.

A **field** is a commutative division ring.

Example: $\mathbb{Z}_p$ with p prime is a field.

Theorem

Let $\mathbb{F}$ be a field. The polynomial ring $\mathbb{F}[x]$ is a Euclidean domain. Specifically, if a(x) and b(x) are two polynomials in $\mathbb{F}[x]$ with b(x) nonzero, then there are unique q(x) and r(x) in $\mathbb{F}[x]$ such that $a(x) = q(x)b(x) + r(x)$ with $r(x) = 0$ or $degr(x) < deg(b(x)$.

Proof

- If $a(x) = 0$, then we take $q(x) = r(x) = 0$ and we have $a(x) = q(x)b(x) + r(x)$
- If $a(x) \neq 0$

  let $deg(a(x)) = n$ and $deg(b(x)) = m$, if $n < m$, then we take $q(x) = 0$ and $r(x) = a(x)$ and we have $a(x) = 0.b(x) + a(x)$, otherwise i.e $n \geq m$, we have:

  $a(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x + a_0$ and $b(x) = b_n x^n + b_{n-1}x^{n-1} + ... + b_1 x + b_0$

  Then the polynomial $a'(x) = a(x) - \frac{a_n}{b_m} x^{n-m} b(x)$ is of degree less than n.

  By induction then, there exists polynomials $q'(x)$ and $r(x)$ with $a'(x) = q'(x)b(x) + r(x)$ with $r(x) = 0 or degr(x) < degb(x)$

  Then, letting $q(x) = q'(x) + \frac{a_n}{b_m} x^{n-m}$ we have

  $a(x) = q(x)b(x) + r(x)$ with $r(x) = 0$ or deg $r(x) <$ deg $b(x)$

Definition 3.1.1.5. Let $g(x), h(x) \in \mathbb{Z}_p[x]$, where no both are zero. Then, **greatest common divisor** of g(x) and h(x) denoted by gcd(g(x),h(x)) is the the unique monic polynomial d(x) $\in \mathbb{Z}_p[x]$ for which:

- d(x) divides both g(x) and h(x).
- if r(x) $\in \mathbb{Z}_p[x]$ and $r(x)|g(x)$, and $r(x)|h(x)$ then $r(x)|d(x)$

Theorem Let $\mathbb{F}$ be a field and let f,g $\in \mathbb{F}[x]$ be any two nonzero polynomials. For every polynomial d$\in \mathbb{F}[x]$, the following properties are equivalent.

- (1) The polynomial d is the gcd of f and g
- (2) The polynomial d divides f and g and there exists u,v $\in \mathbb{F}[x]$ such that $d = uf + vg$

Proof

Let d=gcd(f,g), then $d|f \rightarrow \exists \alpha \in \mathbb{F}[x]$ such that $f = \alpha d$, similarly d divides g and $g = \beta d$, thus $f + g = (\alpha + \beta)d$ then $d = \lambda f + \lambda g$

Now let assume $\exists u, v \in \mathbb{F}[x]$ such that d=uf +vg, then if $\exists h \in \mathbb{F}[x]$ such that h divides both u and v, then h must divide d, thus d=gcd(f,g).

Definition 3.1.1.6. If $f(x) \in F[x]$ where F is a field, then a root of $f(x)$ in F is an element a$\in F$ with $f(a) = 0$

Lemma

Let F be a field and let $f(x) \in F[x]$. Then,for any $a \in F$, there exists $q(x) \in F[x]$ such that $f(x) = (x - a)q(x) + f(a)$

Proof

By the division algorithm, we have $f(x) = (x-a)q(x)+r(x)$ where deg(r)<deg $(x-a) = 1$, and therefore deg(r)=0,i.e $r(x) = r$ is a constant.

So, $f(x) = (x - a)q(x) + r \Rightarrow f(a) = (a - a)q(a) + r = r = r(x)$

Definition 3.1.1.7. Let R be a ring. An *ideal* of R is a non-empty subset I of R with the properties:

i)a,b $\in I \Rightarrow a - b \in I$

ii)$a \in$ I and r $\in R \Rightarrow ra \in I$

Definition 3.1.1.8. A *principal Ideal* I is an ideal generated by a single element. Let R be a commutative ring(and x $\in$ R, the principal ideal generated by x is the set $< x >= \{rx \quad | x \in R\}$

Example 3.1.1.3. I=$< 2 >$, is the set of vectors with all even coefficients.

Example 3.1.1.4. If p is prime, then the ring $\mathbb{Z}_p[x]$ obtained by reducing $\mathbb{Z}[x]$ modulo the prime ideal $< p >$ is a principal ideal domain, since the coefficients lie in the field $\mathbb{Z}_p[x]$.

Definition 3.1.1.9. Let R and S be rings :

A ring homomorphism is a map $\phi : R \to R$ satisfying :

(i) $\phi(a+b) = \phi(a) + \phi(b)$

(ii)$\phi(ab) = \phi(a)\phi(b) \quad \forall a, b \in \mathbb{R}$

3.1.2 Lattices

Studies on lattices have been going on since the late eighteenth century by mathematicians such as Lagrange, Gauss and later Minkowski. In 1996, Miklos showed in a seminal result the use of lattices as cryptography primitives. He generated hard instances of lattices problems which have been used as buildings blocks for lattice-based public-key cryptosystems. We will start by studying in detail some properties of lattices and we will present some underlying hard problems. Then we will present two lattice-based cryptosystems.

Definition 3.1.2.1. A nonempty set $X \subset \mathbb{R}^n$is a vector space if $x + y \in X$ and $cx \in X \forall x, y \in X$ and for all scalars c in $\mathbb{Z}$.

If $x_1, x_2, ..., x_n \in \mathbb{R}^n$ and $c_1, c_2, ...c_k$ are scalars, the linear combination

$$c_1x_1 + c_2x_2 + ... + c_kx_k$$

is independent if

$$c_1x_1 + c_2x_2 + ... + c_kx_k = 0 \Rightarrow c_1 = c_2 = ...c_k = 0$$

Then we say $\{x_1, x_2, ...x_k\}$ is a set of **linearly independent vectors**$\in \mathbb{R}^n$

Let $b_1, b_2, ..., b_n$ be n linearly independent vector $\in \mathbb{R}^n$, **the lattice L** in $\mathbb{R}^n$ generated by them is defined as L=L$(b_1, b_2, ..., b_n)$=$\{\sum_{i=1}^{n} x_i b_i, x_i \in \mathbb{Z}, \forall 1 \leq i \leq n\}$, the set of all linear integer combinations of $b_1, b_2, ..., b_n$

Equivalently , if we define B as the $m \times n$ matrix whose columns are $b_1, b_2, ..., b_n$, then the lattice generated by B is L=L(B)=L$(b_1, b_2, ..., b_n) = \{Bx | x \in \mathbb{Z}_n\}$ Any linearly independent set of vectors that generates L is **a basis** for L. Every lattice has an infinite number of

lattice bases. We say that the **rank of the lattice is n** and its **dimension is m**. If n=m, the lattice is called a **full-rank lattice.**

Example The lattice generated by $\{(1,0)^T,(0,1)^T\}$ is $\mathbb{Z}^2$, the lattice of all integers points. Another basis of $\mathbb{Z}^2$ is $\{(1,0)^T,(0,1)^T\}$, but $\{(1,1)^T,(2,0)^T\}$ is not a basis for $\mathbb{Z}^2$

Definition 3.1.2.2. Let U $\in \mathbb{Z}^{n\times n}$, with the property that $\pm det(U)$, then we say that U is unimodular.

Lemma Two $n \times m$ matrices B and B' generate the same lattice L iff and only if B and B' are related by a unimodular matrix, i.e B'=UB where U is a $n \times n$ matrix.

Proof

Let assume L(B)=L(B'), then for each of the n columns b_i of B', $b_i \in L(B)$. This implies that there exists an integer matrix U $\in \mathbb{Z}^{n\times n}$ for which B'=BU. Similarly, there exists V $\in \mathbb{Z}^{n\times n}$ such that B=B'V. Hence B'=BU=B'VU , and we get ${B'}^T B = (VU)^T {B'}^T B'(VU)$. Taking determinants, we obtain that $det({B'}^T B) = det(VU)^2 det({B'}^T B')$ and hence det(V)det(U)= ± 1. Since V, U are both integer matrices, this means that $det(U) = \pm 1$

Now let assume that B'=BU for some unimodular matrix U. Therefore each column of B' is contained in L(B) and we get $L(B') \subseteq L(B)$. In addition , $B = B'U^{-1}$, and since U^{-1} is unimodular, we get that $L(B') \subseteq L(B)$. We conclude that L(B)=L(B').

Since every lattice has an infinite number of bases, but not every set of n linear independent vectors in $\mathbb{Z}^n$ is a basis of $\mathbb{Z}^n$. How can we tell that a given set of vectors forms a basis for a lattice?

Definition 3.1.2.3. Let L be a lattice of dimension n, and let $b_1, b_2, ..., b_n$ be a basis for L.The fundamental domain (or fundamental parallelepiped) for L corresponding to this basis is the set $P(b_1, b_2, ..., b_n) = \{t_1 b_1 + t_2 b_2 + ... + t_n b_n | x \in \mathbb{R}^n, \forall i : 0 \leq t_i < 1\}$

For a lattice basis B we define the *half open fundamental parallelepiped* for a lattice basis B to be $P(B) = \{Bx | x \in \mathbb{R}^n, \forall i : -\frac{1}{2} \leq x_i < \frac{1}{2}\}$

Lemma

Let L be a lattice of rank n, then $b_1, b_2, ..., b_n$ form a basis of L iff $P(b_1, b_2, ..., b_n) \cap L = \{0\}$

Proof

First, let assume that $b_1, b_2, ..., b_n$ form a basis of L. Then by definition, L is the set of all their integer linear combinations. Since $P(b_1, b_2, ..., b_n)$ is defined as the set of all linear combinations of $b_1, b_2, ..., b_n$ with coefficients in $[0, 1)$, the intersection of the two sets is $\{0\}$ Now let assume, that $P(b_1, b_2, ..., b_n) \cap L = \{0\}$. Since L is a rank n lattice, and $b_1, b_2, ..., b_n$ are linearly independent, we can write any lattice vector $x \in L$ as $\sum y_i b_i$ for some $y_i \in \mathbb{R}$. Since, by definition a lattice is closed under addition, the vector $x' = \sum(y_i - [y_i]b_i)$ is also in L. Then from our assumption, $x' = 0$. This implies that all y_i are integers and hence x is an integer combination of $b_1, b_2, ..., b_n$.

Let L(B) be a lattice of rank n. We define the determinant of L denoted det(L), as the volume of the fundamental parallelepiped, it is also equal to the determinant of any basis of L, namely vol(P(B)=det(L)=$\sqrt{B^T B}$

In our context of lattice-based cryptography, there are "good" and "bad" bases of a lattice. A basis B is said to be good, if the vectors are short and nearly orthogonal. For any basis B, it is true that $\prod_{i=1}^{n} ||b_i|| \geq det(L)$

That means, for a good base we have $\prod_{i=1}^{n} ||b_i|| \sim det(L)$

Hermite Normal Form

Given a square n$\times$ n non singular integer matrix , there exists an n$\times$ n unimodular matrix U and n$\times$ n matrix H such that $AU = H$. H is a lower triangular and is called the Hermite Normal Form (HNF) of A:

i) $h_{ij} = 0 for j > i$

ii) $h_{ii} > 0 \quad \forall$ i and,

iii)$h_{ij} \leq 0$ and $|h_{ij}| < h_{ii}$

Lemma: If B is the basis matrix of a lattice L, then **HNF(B)** is also a basis matrix for L.

The HNF of a basis is unique and can be computed in polynomial time.

Definition 3.1.2.4. Let consider $\vec{v} = (v_0, v_1, ..., v_n)^T$ of a polynomial $v(x) \in R$, we define *the cyclic rotation* denoted by **rot**$(\vec{v})$, and rot$(\vec{v}) := (-v_{n-1}, v_0, ..., v_{n-2})^T$
and its *corresponding circulant matrix***Rot(v)**= $(\vec{v}, rot(\vec{v}), ..., rot^{n-1}(\vec{v})^T$, Rot$(\vec{v})$ is also called the rotation basis of the ideal lattice $< u >$.

Example

let $v(x) = 5x^3 + 3x^2 + 1$, then rot$(\vec{v}) = (-5, 1, 3)^T$ and $Rot(v) = \begin{bmatrix} 1 & 3 & 5 \\ -5 & 1 & 3 \\ -3 & 5 & 1 \\ -1 & 3 & 5 \end{bmatrix}$

3.1.3 Lattice-based cryptosystems

Let us describe the two following hard problems in lattices, used as security assumptions in the lattice-based cryptosystems.
For every n-dimensional lattice L, and i=1,2,...n, the i^{th} successive minimum $\lambda_i(L)$ is the smallest radius r such that Ball(0,r)contains i linearly independent lattice vectors.
The shortest vector problem (SVP) is defined as: given a lattice L, find the nonzero lattice vector $\vec{v}$ closest to the origin ($||\vec{v}|| \leq \gamma\lambda_1(L)$), where γ is a factor depending on the dimension of the matrix. This problem is considered particularly hard to solve for $\gamma \geq n^k, withk > 5$.

Let $dist(L, t) = min_{\vec{v}\in L}\{||t-v||\}$ denote the minimum distance from the target vector t to the lattice L.
For a vector $\vec{v} \in \mathbb{R}^n$, $\vec{v}$ (mod B) is the **unique vector** $\vec{v''} \in P(B)$ so that $\vec{v} - \vec{v''} \in L$, that is $\vec{v}$ (mod B) $= B[B^{-1}\vec{v}]$ where [.] denotes the distance between the coefficients of a vector and the nearest integers.

The closest vector problem (CVP) is defined as: Given lattice L and target point t, find lattice vector $\vec{v}$ closest to $\vec{t}$: $||\vec{v} - \vec{t}|| \leq \gamma$ dist(L,t).

During the mid-1990's, many cryptosystems were introduced whose underlying hard problem was **SVP** and | or **CVP** in a lattice of large dimension. The most interesting of those were the **GGH**crypstosystem of Goldreich, Goldwasser and Silverman. The motivation for introducing these cryptosystems was mainly due to the fact that lattice-based cryptosystems are frequently much faster than factorization or discrete logarithm based ones. RSA for example, would require $O(k^3)$ operations to achieve k bits of security, while encryption and decryption for lattice based require only $O(k^2)$ operations. Lattice-based cryptosystems can also be associated with rings, allowing naturally additions and multiplications.

The GGH cryptosystem works as follow : Alice's private key is a good basis for a lattice L and her public key is a bad basis B_{bad} for L. Bob's message is a binary vector m, which he uses to form a linear combination $\sum m_i b_i^{bad}$ of the vectors in B_{bad}. He then pertubes the sum by adding a small random vector r. The resulting vector differs from a lattice vector by the vector r. Since Alice knows a good basis for L, she can use Babai's algorithm to find v, and then she expresses v in terms of the bad basis to recover m. Trudy in the other hand, knowing only the bad basis B_bad, would not be able to solve the closest vector problem in L.

The N^{th} degree truncated polynomial ring (**NTRU**) is another important lattice-based cryptosystem based on polynomial rings. Let N, p,q be integers such that gcd(N,q)=gcd(p,q)=1 and let define the following $R = \mathbb{Z}[x]/x^N - 1$, $R_p = \mathbb{Z}_p[x]/x^N - 1$, $R_q = \mathbb{Z}_q[x]/x^N - 1$.

Definition 3.1.3.1. For any positive integers d_1 and d_2 we let $T(d_1, d_2) = \{a(x) \in R : a(x)$ has d_1 coefficients equal to 1, a(x) has d_2 coefficients equal to -1, a(x) has all the other coefficients equal to 0$\}$.

Alice chooses public key (N, p,q,d). Alice's private keys consists of two randomly

chosen polynomials $f(x) \in T(d+1,d)$ and $g(x) \in T(d,d)$. Alice computes the inverse $F_q(x) = f(x)^{-1} \in R_p$. Then she computes $h(x) = F_q(x) \times g(x) \in R_q$

. The polynomial $h(x)$ is Alice's public key and the pair $(f(x), F_p(x))$ is her private key. Bob chooses a polynomial $m(x) \in R$ whose coefficients $\in (-\frac{p}{2}, \frac{p}{2})$, he also chooses a random polynomial $r(x) \in T(d,d)$. He, then computes $e(x) = (ph(x) \times r(x) + m(x)) \bmod q$. Alice started the decryption by computing $a(x) = f(x) \times e(x) \bmod q$, then she computes $b(x) = F_p(x) \times a(x) \bmod p$ and recover the message $m(x) = b(x)$.

In the next section, we will describe the first FHE invented by Gentry. The construction has many parameters:

γ is the bit-length of the integers in the public key

Theorem: Babai's closest vector algorithm

Let $L \subset \mathcal{R}^n$ be a lattice with basis $b_1, b_2, ..., b_n$ and let w $\in \mathbb{R}^n$ be an arbitrary vector. If the vectors in the basis are sufficiently orthogonal to another, then the following solves the "Closest Vector problem":

write $w = t_1 v_1 + t_2 v_2 + ... + t_n v_n$ with $t_1, ..., t_n \in \mathbb{R}$

set $a_i = [t_i]$ for i=1,2,...,n

η is the bit-length of the secret key

ρ is the bit-length of the noise

τ is the number of integers in the public key

3.1.4 Gentry FHE

Craig Gentry proposed in 2009, the first fully homomorphic encryption based on ideal in lattices. The Key generation algorithm takes as inputs a fixed ring R and a basis B_i. The public key consists of a "bad" basis B_{pk} of an ideal lattice J, along with some basis B_I of a small ideal I. A ciphertext is a vector close to a J-point, with the message being embedded in the distance to the nearest lattice point. The plaintext space is $R/I = \{0,1\}^n$, for a

message $\vec{m} \in \{0,1\}^n$, we set $\vec{e} = 2\vec{r} + \vec{m}$ for a random small vector $\vec{r}$ and then output the ciphertext $\vec{c} \leftarrow \vec{e} \pmod{B}_{pk}$. We will be working with R, the ring of integer polynomials modulo $f_n(x)$, i.e $R := \mathbb{Z}_p[x]/f_n(x)$ where $f_n(x) = x^n + 1$ with n a power of two. We will also consider the principal ideal I of R. We will denote by $\vec{u} = (u_0, u_1, ..., u_n)^T$, the coefficient vectors of u $\in R$

Generation of the keys of SHE

- Bob chooses a random polynomial $u(x) = \sum_{i=0}^{n-1} u_i x^i \in \mathbb{Z}[x]$, where each entry is a η-bit integer, and he computes p=det (Rot(u(x))should be an odd integer.

- He computes $d(x) = gcd(u(x), f_n(x))$ over $R_p[x]$, and he finds α the unique root of d(x).

  Then he finds a polynomial $v(x) = \sum_{i=0}^{n-1} v_i x^i \in \mathbb{Z}[x]$ such that $u(x) \times v(x) = p \bmod f_n(x)$

  Remark: Not all polynomial $u(x) \in \mathbb{Z}[x]$ will work. We required that u(x) should be such that $HNF(J) = Rot(u(x))$ has the following form:

$$HNF(J) = \begin{bmatrix} p & 0 & 0 & ... & 0 \\ -\alpha & 1 & 0 & ... & 0 \\ -\alpha^2 \bmod p & 0 & 1 & ... & 0 \\ ... & ... & ... & & \\ -\alpha^{n-1} \bmod p & 0 & 0 & ... & 1 \end{bmatrix}$$

- The public key is $pk = (p, \alpha)$, and the secret key is $sk = (p, v(x))$

Now Bob is ready to encrypt his message and send it to Alice.

Encryption of the message

Bob chooses a small random polynomial (noise) and a message $m \in \{0,1\}$, then he computes $c = (2r(\alpha) + m) \bmod p$

3.1.5 Smart-Vercauteren

In 2010, Nigel Smart and Frederick Vercauteren proposed a variant of Gentry scheme. Their scheme is also based on lattices, but they succeeded in reducing the size of the ciphertext and they length of the key. Their scheme also allows efficient fully homomorphic encryption over any filed of characteristic two.

Generation of the keys of SHE

- Bob chooses a random polynomial $u(x) = \sum_{i=0}^{n-1} u_i x^i \in \mathbb{Z}[x]$, where each entry is a η-bit integer, and he computes p=det (Rot(u(x))should be an odd integer.

- He computes $d(x) = gcd(u(x), f_n(x))$ over $R_p[x]$, and he finds α the unique root of d(x).

  Then he finds a polynomial $v(x) = \sum_{i=0}^{n-1} v_i x^i \in \mathbb{Z}[x]$ such that $u(x) \times v(x) = p \bmod f_n(x)$

  Remark: Not all polynomial $u(x) \in \mathbb{Z}[x]$ will work. We required that u(x) should be such that $HNF(J) = Rot(u(x))$ has the following form and moreover p should be a prime:

$$HNF(J) = \begin{bmatrix} p & 0 & 0 & ... & 0 \\ -\alpha & 1 & 0 & ... & 0 \\ -\alpha^2 \bmod p & 0 & 1 & ... & 0 \\ ... & ... & ... & & \\ -\alpha^{n-1} \bmod p & 0 & 0 & ... & 1 \end{bmatrix}$$

- He computes $\beta = (v(x) mod(x)) \bmod (2p)$

  Now his public key is $pk = (p, \alpha)$, and the secret key is $sk(p\beta)$

Now Bob is ready to encrypt his message and send it to Alice.

Encryption of the message

Bob chooses a small random polynomial (noise) and a message $m \in \{0, 1\}$, then he computes $c = (2r(\alpha) + m) \bmod p$ (Notice: The encryption is done bit by bit.)

Decryption of the message

Bob decrypts the ciphertext by computing: $m = (c - [c \times \beta/p + \frac{1}{2}]) \bmod 2$.

Alice can work on the encrypted values now, let define the addition and multiplication

Addition

Given the public key pk, and two ciphertexts c_1 and c_2 : $c = c_1 + c_2 \bmod p$

Multiplication

Given the public key pk, and two ciphertexts c_1 and c_2 : $c = c_1 \times c_2 \bmod p$.

Let illustrate it by an example:

let $n = 2^2 = 4$, and $u(x) = 2x^3 + 4x^2 + 8x + 159$ and $f_n(x) = x^4 + 1$

Generation of the keys

p=det($Rot(u(x))$ = $\begin{bmatrix} 159 & 8 & 4 & 2 \\ -2 & -159 & 8 & 4 \\ -4 & -2 & -159 & 8 \\ -8 & -4 & -2 & -159 \end{bmatrix}$ =641407153

Now we want to find $v(x)$ such that $u(x) \times v(x) = p \bmod f_4(x)$ i.e $u(x) \times v(x) = q(x)(x^4 + 1) + p$, we find $v(x) = -40898x^3 - 91520x^2 - 204800x + 4027071$

Now we compute $d(x) = gcd(u(x) = 2x^3 + 4x^2 + 8x + 159, x^4 + 1)$and we find $d(x) = x - 26912186$

and the we find $\alpha = p - 26912186 = 614494967$ (recall α is the root of d(x) modulo p)

so the public key is (641407153,614494967)

Fully homomorphic properties of the scheme

Let c_1 and c_2 be two ciphertexts such that $c_1 = (2r_1(\alpha) + m_1) \bmod p$ and $c_2 = (2r_2(\alpha) + m_2) \bmod p$, and let $m = m_1 + m_2$.

$c = c_1 + c_2 \bmod p \Rightarrow c = (2r_1(\alpha) + m_1) \bmod p + (2r_2(\alpha) + m_2) \bmod p$

$\exists q, q' \in \mathbb{Z}$ such that $c = pq + 2r_1(\alpha) + m_1 + pq' + 2r_2(\alpha) + m_2$,

then we have $c = p(q + q') + m_1 + m_2 + 2(r_1(\alpha) + 2r_2(\alpha)) \Rightarrow c = pq" + m_1 + m_2 + 2r(\alpha)$ with q"=q+q' and $r(\alpha) = r_1(\alpha) + r_2(\alpha)$

$Decrypt(c) = (c - [c \times \beta/p + \frac{1}{2}]) \bmod 2 = (pq" + m_1 + m_2 + 2r(\alpha) - [(pq" + m_1 + m_2 + 2r(\alpha))\beta/p) + \frac{1}{2}]) \bmod 2 = m_1 + m_2$

Now let $c' = c_1 \times c_2 \bmod p$. and $m = m_1 \times m_2$

Similarly we get $Decrypt(c) = m_1 \times m_2$

3.1.6 Fully homomorphic encryption over integers

This encryption proposed by Van Djik and Gentry, is derived from the one suggested by Gentry in 2009. It uses only elementary modular operations. The security of this scheme relies on the difficulty to find approximate integer gcd, that is given a list of integers that are near-multiples of a hidden integer, output that hidden integer.

Let define the following parameters:

γ is the bit-length of the integers in the public key

η is the bit length of the secret key which is the hidden approximate-gcd of all the public key integers.

ρ is the bit length of the noise, that is the distance between the public key elements and the nearest multiples of the secret key

τ is the number of integers in the public key

Description of the scheme

Step 1: Generation of the keys

The secret key is an odd η bit integer , and p $\in [2^{\eta-1}, 2^{\eta})$

Step 2:Encryption

Let m $\in \{0, 1\}$, then $c = Encrypt(m, pk) = pq + 2r + m$ where r is a random integer 2r is smaller than $|\frac{1}{2}|$

Step 3: Decryption We recover the original message by doing $Decrypt(c) = (c \bmod p) \bmod 2$.

In the next chapter we will present our implementation which is a combination of these algorithms.

CHAPTER 4

IMPLEMENTATION

In this chapter we will describe our implementation. Our algorithm are based on the Smart-Vercauteren approach. We have used the Brenner's code and made some modifications and also the Fast LIbrary For Number Theory.

Key generation

The private and public keys we are generating are all prime

Input

The irreducible monic $f_n(x) = x^n + 1$ (irreducible in $\mathbb{Z}[x]$ for n=power of two). Practically, the user enters the value of n.

We generate a random polynomial with the constant term odd with the requirement that resultant $(f_n(x), g(x)) = p$ prime

The next step in finding the key is to apply the xgcd algorithm to $(f_n(x), g(x))$ to find their gcd and finally get its root.

Output

the pair (p, α) and the secret key q

Encryption

The encryption is done as described in Smart-Vercauteren scheme, that is $c = (2r(\alpha) + m) \bmod p$. In the encryption, we use the homomorphic properties of our scheme to make our scheme more secure. Two bits with a same value would have two different encryptions depending on the function we have chose, since Dec(Enc(1+0))=1+0=1 and Dec(Enc(1.1))=1.1=1.

Decryption

The decryption is similar to the decryption of Smart- Vercauteren.

In this implementation we have reduced the size of the key to 512 bits, to make the encryption runs in real time for visualization. It is not practical for real problems life.

The next generation of FHE will allow multi-party computations. Our future work will be to compare our implementation consumption of energy, since it is designed for smart phones, it must be save energy efficiently.

REFERENCES

[1] Bachman, G., *Lecture on the Theory of Numbers*, Springer, 1951.

[2] Dummit, D.and Foote,R. *Abstract Algebra*, Third edition,Wiley,2003.

[3] Galbraith, S., *Mathematics of Public key Cryptography*, Cambridge University Press, New York, 2012.

[4] Rothe, J., *Complexity Theory AND Cryptography*, World Scientific Publishing Co., Singapore, 2005.

[5] Rudin, W., *Principles of Mathematics Analysis*, McGraw Hill, Inc,1976.

[6] Ryabko, B., *Basics of Contemporary Cryptography for IT Practitioners*, Springer, Berlin Heidelberg, 2005.

[7] Shoup, V., *A Computational Introduction to Number Theory and Algebra*, Cambridge University Press,2005.

VITA

Graduate School
Southern Illinois University

Olive Mbianda

olivembianda.24@siu.edu

NASPT, Yaounde
Bachelor of Science, Engineering, February 2011

Research Paper Title:
Fully Homomorphic encryption for wireless network

Major Professor: Dr. K. Spector Dr. K. Akkaya