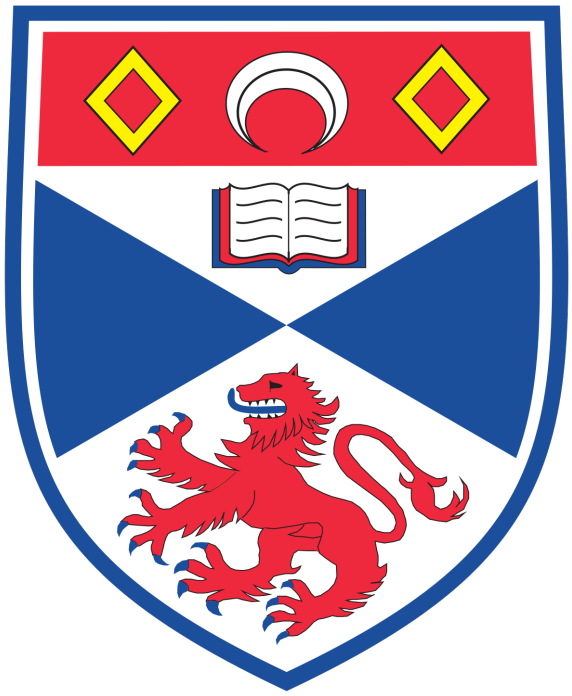
Fully homomorphic encryption

CS4796: Joint Senior Honours Project

Gergely Flamich

Supervisors:
Dr Sophie Huczynska, Prof Steve Linton



School of Computer Science
University of St Andrews
Scotland

Abstract

Contents

1 Introduction

2 A Very Brief History of Ciphers

The history of ciphers goes back at least 2000 years. We will now examine the cipher now named after Julius Ceasar, who used the following method to obfuscate his correspondance for his adversaries:

Ceasar Cipher : Take the message (written using symbols from the Latin alphabet) that we wish to obfuscate. Replace each symbol with the one that comes 3 places after it in the alphabet. For letters at the end where we could not shift, we "wrap around" to the beginning of the alphabet and carry on counting from there. For example

$$\textit{Alea iacta est} \quad \rightarrow \quad \textit{Dohd mdfzd hxz.}$$

To decrypt a message, we simply shift backwards by 3 positions, e.g.

$$\textit{Zhqm, zmgm, zmgm} \quad \rightarrow \quad \textit{Veni, vidi, vici.}$$

We can further generalise this concept to arrive at the definiton of *shift ciphers*, where instead of the fixing the shift to 3, we pick a key $k \in \mathbb{N}$, and we then shift k places forward (and backwards). With shift ciphers in mind, we now formally defined some key concepts that will be used throughout this work.

Definition 2.1. Cipher: Let $\mathcal{M}, \mathcal{K}_{\texttt{Enc}}, \mathcal{K}_{\texttt{Dec}}$ be finite sets of symbols. We will refer to $\mathcal{M}$ as the **message space**. A cipher C over $\mathcal{M}$, $\mathcal{K}_{\texttt{Enc}}$ and $\mathcal{K}_{\texttt{Dec}}$ is a tuple $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$, where

- $\texttt{Gen}$ is the **key generation algorithm**, which outputs a key $(k_{\texttt{Enc}}, k_{\texttt{Dec}}) \in (\mathcal{K}^*_{\texttt{Enc}} \times \mathcal{K}^*_{\texttt{Dec}})$, chosen according to some distribution. We refer to the set $\mathcal{K} \subseteq \mathcal{K}^*_{\texttt{Enc}} \times \mathcal{K}^*_{\texttt{Dec}}$ of all possible outputs of $\texttt{Gen}$ as the **key space** of C. If for all outputs we have $k_{\texttt{Enc}} = k_{\texttt{Dec}}$, then we call C a **symmetric cipher** and instead of the tuple we just write k. Otherwise we call C an **asymmetric cipher**.
- $\texttt{Enc} : \mathcal{K}^*_{\texttt{Enc}} \times \mathcal{M} \rightarrow \mathcal{M}$ is the **encryption algorithm**, which takes a encryption key k and a message m and outputs its encoding c. We will often refer to the message as the **plaintext** and to the encrypted message as **ciphertext**.
- $\texttt{Dec} : \mathcal{K}^*_{\texttt{Dec}} \times \mathcal{M} \rightarrow \mathcal{M}$ is the **decryption algorithm**, which takes some decryption key k and some ciphertext c and outputs its correspnding plaintext m.

Finally, we require the correctness of C, concretely

$$\forall m \in \mathcal{M}, \forall (k, k') \in \mathcal{K}. \quad \texttt{Dec}(k', \texttt{Enc}(k, m)) = m,$$

i.e. that the cipher does not change its contents.

Let us now revisit shift ciphers over an alphabet $\mathcal{M} = \{m_1, \ldots m_n\}$, and see how they fit our definiton of a cipher. First, it is easy to see that we are dealing with a symmetric cipher, and since a shift by k and any $k + xn, x \in \mathbb{N}$ is going to be the same, $\mathcal{K} = \mathbb{Z}_n$. Then,

- $\texttt{Gen}$: $\texttt{Gen}$ is uniformly picking an element from $\mathbb{Z}_n$.

- `Enc`: We note that if we relabel our symbols to their indices (i.e. $m_i \to i$), then we can express our encryption function as
$$\texttt{Enc}(k, i) = i + k \mod n.$$
- `Dec`: Similarly, performing the above relabeling, we get
$$\texttt{Dec}(k, i) = i - k \mod n.$$

Now, checking the correctness of C_{shift}: Let $k, m \in \mathbb{Z}_n$

$$\texttt{Dec}(k, \texttt{Enc}(k, m)) = m + k \mod n - k \mod n = m + k - k \mod n = m \mod n = m.$$

Since k and m were arbitrary, it holds $\forall m, k \in \mathbb{Z}_n$.

Kerckhoffs' principle

3 Perfect Security

4 Computational Security

5 Public-Key Crypto Schemes

5.1 Euler's Theorem

5.2 An Example: RSA

6 Homomorphic Crypto Schemes

6.1 Definition

7 Paillier Encryption

Paillier encryption is an additively homomorphic crypto scheme, that operates over $\mathbb{Z}_{N^2}$ where $N = pq$ for some p, q primes. In particular, it takes advantage of the fact that $\mathbb{Z}_{N^2} \simeq \mathbb{Z}_N \times \mathbb{Z}_N^*$.

8 Fully Homomorphic Encryption

8.1 Lattices

8.2 GGH

8.3 Dr Craig Gentry's Scheme