2007

SECURE IMAGE PROCESSING

Nan Hu
University of Kentucky, nan.hu@uky.edu

ABSTRACT OF THESIS

SECURE IMAGE PROCESSING

In todays heterogeneous network environment, there is a growing demand for distrusted parties to jointly execute distributed algorithms on private data whose secrecy needed to be safeguarded. Platforms that support such computation on image processing purposes are called *secure image processing* protocols. In this thesis, we propose a new security model, called quasi information theoretic (QIT) security. Under the proposed model efficient protocols on two basic image processing algorithms – linear filtering and thresholding – are developed. For both problems we consider two situations: 1) only two parties are involved where one holds the data and the other possesses the processing algorithm; 2) an additional non-colluding third party exists. Experiments show that our proposed protocols improved the computational time significantly compared with the classical cryptographical couterparts as well as providing reasonable amount of security as proved in the thesis.

KEYWORDS: Communication system security, Image Processing, Dis-

tributed Algorithms, Cryptography, Secure Multiparty Computation

---

SECURE IMAGE PROCESSING

By

Nan Hu

Director of Thesis

Director of Graduate Studies

RULES FOR THE USE OF THESES

Unpublished theses submitted for the Master's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgements.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this thesis for use by its patrons is expected to secure the signature of each user.

Name	Date

THESIS

Nan Hu

The Graduate School

University of Kentucky

2007

SECURE IMAGE PROCESSING

THESIS

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in the
College of Engineering
at the University of Kentucky

By

Nan Hu

Lexington, Kentucky

Director: Dr. Sen-ching Samson Cheung , Professor of Electrical and

Computer Engineering

Lexington, Kentucky

2007

ACKNOWLEDGEMENTS

It is my pleasure to express my gratitude to many people without whom this edition would not be possible.

First of all, I would like to express sincere thanks and gratefulness To my advisor and my Thesis Chair, Professor Sen-ching Samson Cheung, Department of Electrical and Computer Engineering, University of Kentucky. I am truly grateful for his dedication to the quality of my research, and his insightful prospectives on numerous prospectives on numerous technical issues.

I would also like to thank members of my Thesis Committee, Prof. Laurence G. Hassebrook and Prof. Yuming Zhang for their valuable suggestions during the final stage of my thesis.

Thanks are also due to the members of MIA Lab, Mr. Vijay Venkatesh. M, Mr. Jian Zhao, Ms. Jayahsri Chaudhari, Mr. Jithendra Paruchuri, and members of the Vis Center, for their help and encouragement.

Finally, I would like to express my deepest gratitude to my parents, for the continuous love, support and patience given to me. Without them, this thesis could not have been accomplished. I am also very

thankful to friends and relatives with whom I have been staying. They never failed to extend their helping hand whenever I went through stages of crisis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

The proliferation of imaging and storage devices and the ubiquitous presence of computer networks make sharing of digital data easier than ever. Such casual exchange of data, however, has increasingly raised questions on how sensitive information can be protected. Consider the scenario in which a user of a cellular-phone camera wants to send his/her pictures to an online photo-processing laboratory for image enhancement such as red-eye removal. The user would be concerned about the privacy of his/her pictures while the online store would need to protect the proprietary enhancement technologies against reverse-engineering. Consider another scenario that a law enforcement agency wants to search for possible suspects in a surveillance video owned by private company A, using a proprietary software from yet another private company B. The three parties involved (agency, company A, company B) all have information they do not want to share with each other (criminal biometric database from the agency, surveillance tape

from company A and proprietary software from company B).

One way of solving this problem is the Trusted Computing (TC) Platform where the software is executed in a secure memory space of the client machine equipped with a cryptographic co-processor [34]. Besides the high cost of overhauling the existing PC platform, the TC concept remains highly controversial due to its unbalanced protection of the software companies over the consumers [3]. To balance the protection for both the clients and the servers, another solution is then proposed by establishing a joint computation and communication platform that can guarantee the secrecy of private data and algorithms, and at the same time achieve a well-defined objective that benefits all parties involved. Platforms that provide security to the joint image processing algorithms are called *secure image processing* protocols.

The secure joint computation aforementioned is, however, not a new problem. Such type of secure computation in a distributed environment is a well-known problem in cryptography, and is referred to as the Secure Multiparty Computation (SMC) problem. The goal of a SMC protocol is to allow multiple distrusted parties jointly compute a function without complete sharing of their own information [15]. Like many other cryptographic protocols, the security of SMC protocols can be guaranteed under two different security models — information-theoretical security and computational security. Information-theoretically secure protocols protects privacy in such a way that the information ex-

changed in the protocol provides no additional information, measured in entropy, about the private data. In computationally secure protocols, private information is first transformed before transmitting to other parties. The security is based on the huge computational burden of performing the inverse transformation. Although the information-theoretic security model provides the ideal level of security, it has been shown that many simple operations like inner product or thresholding cannot be securely computed between two distrusted parties [22]. As a result, most existing SMC protocols are built under the computational security model [38, 15, 4].

There has been little work in applying SMC to image processing problems. The only work known to us is by S. Avidan et. al. [4] on applying classical SMC protocols for two-party face detection. In a typical classification task such as face detection, a significant portion of an image is transformed into feature vectors, which in most cases cannot be used to recover the original image. The manipulation of feature vectors is thus secure by definition and no special SMC protocols are required. As a result, the complex SMC protocols do not significantly affect the overall performance of the classification task. On the other hand, many common image processing applications require pixel-by-pixel processing. The high compuational compexity of most SMC protocols becomes a major drawback and hence useless when applied to pixel level image processing algorithms. For example, the

classical solution to the thresholding problem[1], or comparing two private numbers a and b, is to use Oblivious Transfer (OT) primitive [37] – one party (Bob) creates a series of tables by bitwise comparing b with every possible value of a, encrypts the tables using a public-key cipher, and transfers them to Alice. Alice decrypts the entries in the tables that correspond to his own number a and deduces the result. Most public-key ciphers use modular exponentiations on very large finite field which is complex to compute. As a result, it is difficult to scale these protocols to signal processing applications that requires handling a large amount of data and satisfying the real-time constraint.

As a result, it is imperative to develop fast computation techniques for these applications. Among all image processing techniques, linear filtering and thresholding are arguably the most basic and useful ones. As mathematically simple as they are, they have been used in most of the complex and advanced image processing, computer vision and pattern recognition applications such as enhancement, denoising, halftoning, 3-D reconstruction and varies detection algorithms. Hence, we focus in this thesis on solving the secure linear filtering and thresholding problems only. Even though linear filtering by itself is inherently insecure as we will demostrate in Section 1.3, we expect it when used in combination with other types non-linear processing algorithms such as thresholding to provide security.

[1]This problem is commonly referred to as the Secure Millionaire Problem in SMC literature.

1.2 Contributions

Our major contribution in this thesis is the mathematical formation of a new security model, called Quasi-Information-Theoretic (QIT) security model and the corresponding QIT linear filtering and thresholding protocols which is a key step in building secure pattern recognition applications. The proposed QIT secure model is a framework that is expected to enable the development of more efficient secure image processing protocols besides those developed in this theis. Hence, our work could be deemed as an introduction to a relatively new interdisciplinary research area between security engineering and image processing. The QIT model is a weaker form of information-theoretic security. Its security is provided by using non-invertible transformations on private data. Though not explicitly defined, QIT-secure protocols have already been developed for inner product computation [11]. Compared with existing SMC protocols, our proposed linear filtering protocol provides QIT security to both parties and our thresholding protocol is more secure to one party (Alice) but not as secure to the other (Bob) – Alice can deduce Bob's number to be among n distinct numbers spread through the entire range of the input. n is a design parameter that can be changed based on the target level of security. All our proposed protocols executes significantly faster than existing protocols.

1.3 Problem Description

In this section, we will introduce the problem definitions and some of the notations used throughout this thesis. Specifically, we will explain the reason for linear filtering to be inherently insecure.

1.3.1 Linear Filtering

Given an image $\{x(\mu,\nu) \ : \ 0 \leq \mu \leq N_1, \ 0 \leq \nu \leq N_2\}$ and a filtering operation $f(\cdot)$ described by a set of parameters Θ, we define the output $y(\mu,\nu)$ of applying this filter to x as follows:

$$y(t) = f(x;\Theta) \tag{1.1}$$

In the secure image filtering model, we have two parties, Alice and Bob, who own the signal x and the filter parameter Θ respectively. Our goal is to establish a computation protocol between Alice and Bob so that

1. Alice obtains $f(x;\Theta)$ without any knowledge of Θ, and
2. Bob does not know anything about x.

For linear filtering, the filter parameters are specified as a filter mask $\mathbf{h}$ defined as $\{h(i,j) : -\frac{l_1}{2} \leq i \leq \frac{l_1}{2}, \ -\frac{l_2}{2} \leq j \leq \frac{l_2}{2}\}$. The linear filtering operation can then be written as

$$y(\mu,\nu) = x \otimes h = \sum_{i=-l_1/2}^{l_1/2} \sum_{j=-l_2/2}^{l_2/2} h(i,j)x(\mu-i,\nu-j). \tag{1.2}$$

It is easy to see that Equation (1.2) is a scalar product between two $(l_1+1)(l_2+1)$ dimensional vectors.

Our secure linear filtering protocol use the following conceptual model: Alice first forms a $N_1N_2 \times (l_1+1)(l_2+1)$ matrix X_w whose rows are the signal data needed for the inner product operation. The total number of rows of X_w is the total number of pixels in the output image[2]. If we denote the i^{th} row of X_w as $X_w(i,:)$, the output image as a vector $\mathbf{y} = [y(1,1) \ \cdots \ y(N_1,N_2)]^T$, and the filter mask as a vector $\mathbf{h} = [h(-\frac{l_1}{2},-\frac{l_2}{2}) \ \cdots \ h(\frac{l_1}{2},\frac{l_2}{2})]^T$, then the linear image filtering could be written as

$$\mathbf{y} = X_w\mathbf{h} \tag{1.3}$$

A secure linear filtering protocol decomposes every $y(i) = X_w(i,:)\cdot\mathbf{h}$ into $y(i) = y_a + y_b$ such that Alice computes y_a without any knowledge of $\mathbf{h}$ and Bob computes y_b without any knowledge of $X_w(i,:)$.

If linear filtering is the end goal of the processing, Bob sends back his portion to Alice to compute the output y. Using both the input x and y, Alice can estimate $\mathbf{h}$ using the least square estimate

$$\hat{\mathbf{h}} = (X_w^TX_w)^{-1}X_w\mathbf{y}.$$

In other words, linear filtering is *intrinsically insecure* to Bob no matter how secure the protocol is. General non-linear filtering, on the other hand, is much harder to invert based on limited number of input and

[2] X can be made to have N_1N_2 rows with appropriate boundary handling.

output pairs. If linear filtering is used as part of a secure non-linear processing system, it is thus important that *neither Alice nor Bob has the entire output of the linear filtering protocol.*

1.3.2 Thresholding

Thresholding is secrete comparision, i.e. Alice holds a secret scaler a, and Bob holds another secret scaler b. They want to find out who has a bigger number without disclosing their private data. We propose to convert this problem into a special polynomial evaluation problem. Alice first randomly generate a $(n-1)^{\text{th}}$-degree real polynomial $f(x)$ such that $(f(a) - f(b))(a - b) \geq 0$ for $\forall a, b \in \mathbb{R}$. Without loss of generality, let $f(x)$ takes the form $f(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$. Then it is straightforward to see that b is greater than a if and only if $f(b) > 0$. Thus if Bob knows the value of $f(b)$, he can easily solve the problem without any knowledge of a. To compute $f(b)$, we note that $f(b)$ can be computed as an inner product:

$$f(b) \triangleq a_{n-1}b^{n-1} + \cdots + a_1 b + a_0 \triangleq \mathbf{x}_1^T \mathbf{x}_2 \tag{1.4}$$

where $\mathbf{x}_1 = [\ a_{n-1}\ \cdots\ a_1\ a_0\ ]^T$ and $\mathbf{x}_2 = [\ b^{n-1}\ \cdots\ b\ 1\ ]^T$.

At the end of the protocol, $f(b)$ is separated into two parts $f(b) = r_a + r_b$, where Alice holds the partial result r_a and Bob holds the other partial result r_b. After the protocol, if Bob want to know whether his number is larger, Alice needs to send her portion r_a to Bob, and Bob

compute $f(b)$. Bob can infer the relationship of a and b from the sign of $f(b)$

Although the linear filtering protocol, whatever it is, is *intrisically insecure* as discussed in Section 1.3.1, when we combine linear filtering and thresholding together, such as in a denoising algorithm, this problem will no longer exist. Since the partial outputs of the linear filtering need not to be summed together to conduct a threholding. Assume the threshold Bob has is t_0, and the intention of thresholding is to compare whether $a + b$ is larger than t_0 or not. This is the same as compare a with $t_0 - b$. Thus, what Bob needs to do is to substract b from t_0 and perform the thresholding protocol with Alice to find the relationship between $a + b$ and t_0.

1.4 Organization Of The Thesis

The remaining chapters of the thesis is organized as follows: In Chapter 2, we will review the exisiting related research work. We will then introduce the existing security models before we mathematically form the QIT security model of our own in Chapter 4. In Chapter 5, the security of the proposed linear filtering and thresholding protocols will be proved and comparison of the performance of our protocols with existing SMC protocols will be shown in Chapter 6 together with a brief discussion of several possible weak point of our protocols. Finally, the thesis is concluded in Chapter 7.

Chapter 2

Related Research Work

In this chapter, we will review several related area about the secure image processing project. One is the cryptographic counterpart, i.e. the secure multiparty computation (SMC) and the oblivious transfer (OT) primitive. Another is the previous work that have applied the SMC protocols on image processing algorithms. The third is the existing research work on the protocols that satisfies our proposed quasi information theoretic (QIT) security. We will review these in separated sections.

2.1 SMC and OT

The general problem of secure multiparty computation (SMC) can be traced back to the classical paper by Yao [37]. In that paper, he introduced the millionaire problem and it was then further extended by Goldreich, Micali, and Wigderson [16] and many others to form the concept of SMC. In a general setting of a SMC protocol [15], we have a given number of participants $p_1,\ p_2,\ \cdots,\ p_N$, each having a

private data, respectively $d_1,\ d_2,\ \cdots,\ d_N$. The participants want to compute the value of a public function F on N variables at the point $(d_1,\ d2,\ \cdots,\ d_N)$. A SMC protocol is dubbed secure if no participant can learn more from the description of the public function and the result of the global calculation than what he/she can learn from his/her own entry - under particular conditions depending on the model used.

There are basically two types of security models for a SMC protocol as briefly introduced in Chapter 1. One is called computational security, which is based on the hardness of some mathematical problem, like factoring and discrete logarithm. The other one, unconditional security which is often referred to as information theoretic security, is usually with some probability of error which can be made arbitrarily small. Different SMC protocols have been developed under both models [8, 23, 29, 35]. The assumptions used in a SMC prtocol could be that participants use a synchronised network (a message sent at a "tick" always arrives at the next "tick"), that a secure and reliable broadcast channel exists, that a secure communication channel exists between every pair of participants (an adversary cannot read, modify or generate messages in the channel), etc.. The centrally controlled adversary considered can be passive (only allowed to read the data of a certain number of participants) or active (can corrupt the execution protocol or a certain number of participants). An adversary

can be static (chooses its victims before the start of the multiparty computation) or dynamic (can chose its victims during the course of execution of the multiparty computation). Specially, the protocol is said to be secure in a *semi-honest* environment if all parties respect the protocol and are not able to derive more information than what can be deduced fro mthe final results. While most of the SMC protocols, including those descibed in this theis, are developed under this assumption, there some work extending the assumption to malicious environment when participants can do whatever to know as much as possible [10, 18, 13, 9, 25].

One of the basic tools used in Perfectly Secure Multiparty Computation (PSMC) is *secret sharing*. A t-out-of-m secret-sharing scheme breaks a secret number x into m *shares* $r_1, r_2, \ldots, r_m$ such that x cannot be reconstructed unless an adversary obtains more than $t-1$ shares with $t \leq m$. The importance of a secret-sharing scheme in PSMC is illustrated by the following example: in a 2-party secure computation of $f(x_1, x_2)$, party P_i will use a 2-out-of-2 secret-sharing scheme to break x_i into r_{i1} and r_{i2}, and share r_{ij} with party P_j. Each party then computes the function using the shares received, resulting in $y_1 \triangleq f(r_{11}, r_{21})$ at P_1 and $y_2 \triangleq f(r_{12}, r_{22})$ at P_2. If the secret sharing scheme is *homomorphic* under the function $f()$, that is y_1 and y_2 are themselves secret shares of the desired function $f(x_1, x_2)$, $f(x_1, x_2)$ can then be easily computed by exchanging y_1 and y_2 between the two par-

ties. Under our computational model, all SMC problems can be solved if the secret-sharing scheme is *doubly homomorphic* – it preserves both addition and multiplication. Adi Shamir [31] invented such a t-out-of-m scheme called *Shamir's Secrety Sharing* scheme. In this scheme, the secrete number x is hidden as the constant term of randomly generately polynomial of degree $t-1$ and the values of the polynomial evaluated at m different points are distributed among the m participating parties. To recover the secret, i.e. the constant term, at least t parties need to share their information.

It is unsatisfactory that PSMC cannot even provide secure two-party computation [22]. Instead of relying on perfect security, modern cryptographical techniques primarily use the so-called *computational security* model. Under this model, secrets are protected by encoding them based on a mathematical function whose inverse is difficult to compute without the knowledge of a secret key. Such a function is called *one-way trapdoor function* and the concept is used in many public-key cipher: a sender who wants to send a message m to party P will first compute a ciphertext $c = E(m, k)$ based on the publicly known encryption algorithm $E()$ and P's advertised public key k. The encryption algorithm acts as a one-way trapdoor function because a computationally-bounded eavesdropper will not be able to recover m given only c and k. On the other hand, P can recover m by applying a decoding algorithm $D(E(m, k), s) = m$ using her secret key s.

Unlike perfectly secured protocols in which the adversary simply does not have any information about the secret, the adversary in the computationally secured model is unable to decrypt the secret due to the computational burden in solving the inverse problem. Even though it is still a conjecture that true one-way trapdoor functions exist and future computation platforms like quantum computer may drastically change the landscape of these functions, many one-way function candidates exist and are routinely used in practical security systems [1].

One important SMC primitive under the computational security model is oblivious transfer (OT), which is a protocol by which a sender sends some information to the receiver, but remains oblivious as to what is sent. The first form of oblivious transfer was introduced in 1981 by M. Rabin [30]. Even et al. [12] then extended the work and developed a more useful form of OT, called "1 out of 2 oblivious transfer" and denoted by OT_1^2, for the purpose of SMC. Later, the concept of OT is further extended to "1 out of n oblivious tranfer" denoted by OT_1^n in [28, 2]. Further work has revealed oblivious transfer to be a fundamental and important problem in cryptography. It is considered one of the critical problems in the field, because of the importance of the applications that can be built based on it. In particular, it is a 'complete' for secure multiparty computation: that is given an implementation of oblivious transfer it is possible to securely evaluate any polynomial time computable function without any additional primi-

[1] A list of one-way function candidates can be found in [17, ch.1].

tive [21]. Because of the importance of OT in Cryptography as well as in this thesis, the OT_1^2 protocol is detailed in Algorithm 1 [4] as an example of OT.

Algorithm 1 $\mathtt{OT}_1^2()$

Require: Alice has $\sigma \in \{0, 1\}$, and Bob has two messages $M_0,\ M_1$.

1: Bob sends Alice two different public encryption keys K_0, K_1.
2: Alice generates a key K and encrypts it with K_0 or K_1 according to her σ. Without loss of generality, let's say $\sigma = 1$. She sends Bob $E(K, K_1)$, i.e. she encrypts K with Bob's K_1 since $\sigma = 1$.
3: Bob does not know which public key Alice used, so he decrypts with both of his private keys. He thus obtains both the real key K, and a bogus one K'.
4: Bob sends Alice $E(M_0, K')$ and $E(M_1, K)$, in the *SAME* order he sent the keys K_0 and K_1 in step 1. Alice decrypts the second of these messages with the key K and obtains M_1.
5: **return** Alice knows M_1.

Let's consider the security here. Can Alice know more than M_1? She would need to know K' which requires the knowledge of K_0. Can Bob know which one Alice has selected? He would need to differentiate K and K' and find which one is the real key. But these two keys both look like random strings.

While most of the above-mentioned results are established in 1980s, SMC continues to be a very active research area in cryptography and its applications begin to appear in many other disciplines. Recent advances focus on better understanding of the security strength of individual protocols and their composition, improving CSMC protocols in terms of their computation complexity [28, 26] and communication cost [6, 27, 1, 5], relating SMC to error correcting coding [14, 33], and introducing SMC to a variety of applications [24, 11, 7, 4, 19, 20].

2.2 Applications of SMC in image processing

The only work known to us in the application of SMC protocols on image processing algorithms is by S. Avidan et al. [4]. They converted the Viola-Jones type face detector [36] by rewriting the detector using cryptographic primitives. Basically, a Viola-Jones type face detector is a AdaBoost based detector combining a series of weak classifiers in a 'Cascade' manner to form a strong classifier. Mathematically, the weak classifiers are built on vector inner product between feature vector extracted and the classifier parameters followed by a thresholding on the result. The only mathematical computation involved here are vector inner product and thresholding. Hence, SMC protocols for vector inner product and thresholding are formed by utilizing the oblivious transfer (OT) primitive. In addition, the authors incorperate image hashing by histograms of oriented gradients to accelerate the processing.

Since the feature vectors were extracted before the classifiers were applied, the feature extraction process was not involved in the secure detection stage, which tremendously reduced the computation burden. However, as in our problems of image filtering and thrsholding, the computation is on pixel-by-pixel basis. The sheer number of pixels in common images requires far more computation than simply the classifiers do. Therefore, it is impossible for us to employ the methods used in [4].

2.3 QIT Secure Inner Product

Du et. al. proposed a vector inner product protocol in 2004 [11], which is the only known exsting inner product protocol that satisfies our proposed security model. We briefly describe this protocol using the pseudo-code `InnerProductAlice` and `InnerProductBob` listed below. Alice has a m-dimensional vector $\mathbf{x}$ and Bob has a m-dimensional vector $\mathbf{y}$. They both know an invertible matrix P and its inverse P^{-1}. P is broken down into top and bottom halves $T \in \mathbb{R}^{\lfloor \frac{n}{2} \rfloor \times n}$ and $B \in \mathbb{R}^{(n-\lfloor \frac{n}{2} \rfloor) \times n}$, while P^{-1} into left and right halves $L \in \mathbb{R}^{n \times \lfloor \frac{n}{2} \rfloor}$ and $R \in \mathbb{R}^{n \times (n-\lfloor \frac{n}{2} \rfloor)}$. The inner product $\mathbf{x}^T\mathbf{y}$ can then be decomposed as follows:

$$\mathbf{x}^T\mathbf{y} = \mathbf{x}^T P^{-1} P\mathbf{y} = \mathbf{x}^T LT\mathbf{y} + \mathbf{x}^T RB\mathbf{y} \tag{2.1}$$

Alice then sends $\mathbf{x}^T R$ to Bob who computes $\mathbf{x}^T RB\mathbf{y}$ while Bob sends Alice $T\mathbf{y}$ so that she can compute $\mathbf{x}^T LT\mathbf{y}$.

Algorithm 2 `InnerProductAlice`$(\mathbf{x}, P^{-1})$

Require: $\mathbf{x} \in \mathbb{R}^n$. $P^{-1} = (L \quad R)$ is a $n \times n$ invertible matrix where $n \geq 2$; $L \in \mathbb{R}^{n \times \lfloor n/2 \rfloor}$ and $R \in \mathbb{R}^{n \times (n-\lfloor n/2 \rfloor)}$.

1: $\mathbf{x_1} \leftarrow L^T\mathbf{x}$
2: $\mathbf{x_2} \leftarrow R^T\mathbf{x}$
3: Transmit $\mathbf{x_2}$ to Bob.
4: Receive $\mathbf{y_1}$ from Bob.
5: **return** $\mathbf{x_1}^T\mathbf{y_1}$

The security of the protocol comes from the observation that $\mathbf{x}^T R$ and $T\mathbf{y}$ project $\mathbf{x}$ and $\mathbf{y}$ into lower-rank subspaces, and thus the components of the original vectors inside the null spaces of the matrices are

Algorithm 3 `InnerProductBob`$(\mathbf{y}, P)$

Require: $\mathbf{y} \in \mathbb{R}^n$. $P^T = (T^T \quad B^T)$ is a $n \times n$ invertible matrix where $n \geq 2$; $T \in \mathbb{R}^{\lfloor n/2 \rfloor \times n}$ and $B \in \mathbb{R}^{(n-\lfloor n/2 \rfloor) \times n}$.

1: $\mathbf{y_1} \leftarrow T\mathbf{y}$
2: $\mathbf{y_2} \leftarrow B\mathbf{y}$
3: Receive $\mathbf{x_2}$ from Alice.
4: Transmit $\mathbf{y_1}$ to Alice.
5: **return** $\mathbf{x_2}^T\mathbf{y_2}$

irrecoverably lost. In [11], the authors proposed a design of P based on decoding matrices used in error control coding so as to spread the projections of the neighboring vectors as far as possible. Unlike the OT procotol which requires complex long-integer modular exponentiation and random key generation, this protocol requires only the highly optimized matrix multiplications.

Although, technically speaking, linear filtering is just a series of vector inner product, the above proposed method cannot be directly applied. The reason is because of the overlapping between adjacent vectors make the reverse engineering possible, and hence comprised the security as was explained in Chapter 1.

Chapter 3

Security Models

Following the convention used in cryptography, we refer the private information as *plaintext* and the information exchanged among distrusted parties as *ciphertext*. All existing cryptographic protocols are based on one of the two security models – information theoretic security and computational security. In the following sections, the definition of these two security models will be breifly introduced. However, as discussed in Chapter 1, the two models are not suitable for pixel level computation because either there are no solution (for information theoretic security) or it is too computationally expensive to be applied in the pixel level applications (for computational security). Thus, a new security model is proposed here and based on the proposed new security model protocols to solve the linear convolution and thresholding problems are designed and will be discussed in Chapter 4.

3.1 Information Theoretic Security

A cryptosystem satisfies *information theoretic security* if its security derives purely from information theory. That is, it makes no unproven assumptions such as the hardness of mathematical problems such as discrete logarithm, and is hence secure even when the adversary has unbounded computing power [15].

The normally referred information theoretic security is also called *perfect security*. Shannon originally formulated this security, though defined in a different, but equivalent way [32]. Thus, Perfect Secrecy is also sometimes called Shannon Secrecy.

Definition 1. *Let A be a cryptographic protocol, $\mathcal{P}$ be the plaintext set, and $\mathcal{C}$ be the ciphertext set. $\forall\ x \in \mathcal{P}$, let $y \in \mathcal{C}$ be the corresponding ciphertext. Let $P(\cdot)$ be the probability function. Then, A is said to satisfy information theoretic security if*

$$P(x|y) = P(x),$$

From the definition, we could see that information theoretic security means the *a posteriori* probability of the plaintext being x, given that the ciphertext y is observed, is identical to the *a priori* probability of the plaintext being x, i.e. knowing y gives no help in knowing x.

3.2 Computational Security

Unlike the *information theoretic security*, which makes no unproven assumption of the hardness of some mathematical problems, *Computational Security*, also know in the cryptographic society as *Semantic Security* makes necessary assumptions on the hardness of some mathematical problems such as factoring and discrete logarithm for computationally bounded adversaries.

Definition 2. *Let $P(\cdot)$ be the probability function, and $l(n)$ be any polynomial over n. Then, a cryptographic protocol A is said to satisfy computational security if for all polynomial-time algorithm G, and large enough $n_0 \in \mathbb{N}$, $\forall\ n > n_0$,*

$$P(G(y, n) = x) < \frac{1}{l(n)},$$

Computational Security means given the ciphertext y and any public information, no polynomial-time algorithm can compute the correct plaintext x with a non-trivial probability. In another word, a cryptographic protocol is *Computationally Secure*, if it is infeasible or takes forever (long enough time in realistic) for a computationally bounded adversary to derive significant information about the message (plaintext) from the given ciphertext.

3.3 Quasi Information Theoretic (QIT) Security

As popular as the above introduced security models, they are not suitable for pixel level computation tasks as we are dealing with. Instead, we propose our new notion of security which is based on non-invertible mappings. Hence, it is necessary to define non-invertibility first.

Definition 3. *Let $g : \mathcal{X} \to \mathcal{Y}$ be a mapping from a probability sample space*[1] *$\mathcal{X}$ to another probability sample space $\mathcal{Y}$. $\forall\ x \in \mathcal{X}$ with $P(x) > 0$, define $g^{-1} \circ g(x) = \{\ \alpha \mid \alpha \in \mathcal{X},\ g(\alpha) = g(x),\ \text{and}\ P(\alpha) > 0\}$.*

1. *Given $\alpha, \beta \in \mathcal{X}$ with non-trivial probability, they are called QIT-indistinguishable if $g(\alpha) = g(\beta)$.*

2. *Given $x \in \mathcal{X}$ with $P(x) > 0$, $g^{-1} \circ g(x)$ is called the QIT indistinguishable set of x under g.*

3. *g is called* noninvertible*, if the probability of finding a $x \in \mathcal{X}$ whose QIT indistinguishable set has no element besides x is zero, i.e. $P\big(\{\ \alpha \mid \alpha \in \mathcal{X}, |g^{-1} \circ g(x)| < 2\}\big) = 0$. In particular, we call $g(x)$ N-noninvertible if the probability of finding a QIT indistinguishable set smaller than N is zero.*

Notice that given $\alpha \in g^{-1} \circ g(x)$, there is no relative increase in the knowledge about α and x based on $y = g(x)$. This can be easily

[1] We assume the probability space discrete. If it is continuous, then X and Y will be the collection of measurable sets.

shown by using the Bayes rule:

$$\frac{P(x|g(x)=y)}{P(\alpha|g(\alpha)=y)} = \frac{P(g(x)=y|x)P(x)/P(y)}{P(g(\alpha)=y|\alpha)P(\alpha)/P(y)} = \frac{P(x)}{P(\alpha)} \tag{3.1}$$

Any cryptographic protocol A can be viewed as a mapping from the plaintext $\mathcal{P}$ to the ciphertext $\mathcal{C}$. As such, we introduce the following definition:

Definition 4. *A cryptographic protocol A satisfies called QIT security if the underlying mapping A from plaintext space to ciphertext space is non-invertible. A is N-QIT secure if the mapping is N-noninvertible.*

It is obvious that the QIT security is weaker than the information theoretic security as g can be any noninvertible mapping which can certainly provide additional information about the plaintext $x \in \mathcal{P}$ given the ciphertext $y = g(x) \in \mathcal{C}$, i.e. $P(x|y) > P(x)$. On the other hand, based on Equation 3.1, the QIT model guarantees that the relative relationship between two plaintexts x and α that map to the same ciphertext y remains unchanged, though the individual conditional probability may increase.

QIT security is also different from computational security. The classic computational security model depends solely on the computational hardness of computing the plaintext x given the ciphertext $y = g(x)$. However, for a given y, it is guaruanteed that there is only one x that satisfies $g(x) = y$. In QIT security, computing the QIT

indistinguishable set $g^{-1} \circ g(x)$ of x for a given mapping if often quite straightforward. However, the cardinality of $g^{-1} \circ g(x)$ could be large and the true identity of x will remain hidden. It can also be seen from Equation 3.1 that, if $P(\alpha) = P(x)$, then $P(\alpha|g(\alpha)) = P(x|g(x))$, i.e. if the plaintext is uniformly distributed, the *a posteriori* probability is also uniform within the QIT indistinguishable set of x. In this special case, there is no algorithm that can distinguish between α and x.

Chapter 4

Protocols

In this chapter, we will describe our protocols in detail of how to solve the linear filtering and thresholding problem. Here we assume both Alice and Bob are semi-honest, as defined in cryptography, i.e., both parties are going to respect the protocol, but they are curious when the protocol is finished, which means they are going to do whatever to compute the other party's information from what they have recieved during the execution of the protocol. In addition, during the design of the protocols, we assume that Bob is the server or the image processing algorithm provider, who possesses more computational power than Alice which is client or the image holder. As a result, we try to assign the computational jobs to Bob whenever possible as long as it does not detroy the security of the protocols.

4.1 Linear Filtering

In this section, we will develop two types of secure filtering protocol: 1) a two party protocol based on rank deficient matrix transform and

2) a three-party protocol based on random permutation of the data. Before we introducing the new protocols, we first review the classic two-party protocol based on OT, the details of which is introduced in Chapter 2.

4.1.1 Classical Two-party Solution

As introduced in Chapter 2, oblivious transfer allows Alice to select one element from the whole dataset Bob holds without revealing to Bob which element Alice has selected and without knowing any othe element in the dataset rather than the one selected. Thus, a secure scalar product protocol can be implemented based on the above-mentioned property of oblivious transfer and is detailed in Algorithm 4.

Algorithm 4 `OTInnerProd`$(\mathbf{x}, \mathbf{h})$

Require: $\mathbf{x}, \mathbf{h} \in F^m$, F is some finite field and $|F| = M_F$.

1: Bob computes for each h_i a table of M_F entries, where the j-th enthy of the table is $j \cdot h_i - r_i$ and r_i, $1 \leq i \leq m$ are the random numbers generated by Bob and known only to him.

2: Alice and Bob engage in m rounds of $OT_1^{M_F}$ protocols in which Alice selects the j-th entry of the table in the i-th round if $x_i = j$.

3: Alice takes the sum of the m quatities $a = \sum_{i=1}^{m}(x_i \cdot h_i - r_i) = \sum_{i=1}^{m} x_i \cdot h_i - \sum_{i=1}^{m} r_i$.

4: Bob computes the sum of all r_i's, $b = \sum_{i=1}^{m} r_i$.

Alice and Bob each hold a m-dimensional vector, and it is obvious that after the protocol Alice and Bob each hold a number a and b as described respectively and

$$a + b = \sum_{i=1}^{m} x_i \cdot h_i$$

makes sure the correctness of the protocol and the property of Oblivious Transfer and randomness of r_i's guarantee the security of the whole protocol.

4.1.2 QIT Two-party Solution

It may seem intuitive to implement secure linear filtering by applying the inner product algorithm [11] as described in Chapter 2 on X_w row by row. However, it is not secure as adjacent rows in X_w overlap with each other. As a result, the redundancy in the rank-reduced data sent to Bob allows him to form a least-square estimation of the original image. This least square problem involves solving a least-square data matrix of size $N\lfloor\frac{(l+1)}{2}\rfloor \times N$. To achieve the QIT secrecy, Alice and Bob need to carefully designed matrix P. The proposed protocol is described below in Algorithm 5 and 6.

Algorithm 5 `FilterAlice`(X_w, m)

Require: $X_w \in \mathbb{R}^{n\times m}$, which is reformated from the original image.

1: Receive $P^{-1} = (L\ R)$ from Bob.
2: $X_1 \leftarrow X_w L$.
3: $X_2 \leftarrow X_W R$.
4: Transmit X_2 to Bob.
5: Receive $\mathbf{h}_1$ from Bob.
6: **return** $X_1\mathbf{h}_1$

At the end of `FilterAlice` and `FilterBob`, Alice and Bob each hold the quantity $X_1\mathbf{h}_1$ and $X_2\mathbf{h}_2$ respectively. The correctness of the protocol can be easily tested by

$$X_1\mathbf{h}_1 + X_2\mathbf{h}_2 = X_w LT\mathbf{h}_1 + X_w RB\mathbf{h}_2 = X_w P^{-1}P\mathbf{h} = X_w\mathbf{h}. \quad (4.1)$$

Algorithm 6 `FilterBob`$(\mathbf{h}, m)$

Require: $\mathbf{h} \in \mathbb{R}^{m\times 1}$.

1: Generate matrix $L \in \mathbb{R}^{m\times\lfloor\frac{m}{2}\rfloor}$ and form $P^{-1} = (L\ R) \in \mathbb{R}^{m\times m}$ where $R\perp L$. Computer $P^T = (T^T\ B^T)$ under the constrain $PP^{-1} = I$, where I is the identity matrix.
2: Send Alice the matrix P^{-1}.
3: $\mathbf{h}_1 \leftarrow T\mathbf{h}$.
4: $\mathbf{h}_2 \leftarrow B\mathbf{h}$.
5: Receive X_2 from Alice.
6: Transmit $\mathbf{h}_1$ to Alice.
7: **return** $X_2\mathbf{h}_2$

Algorithm 5 and 6 may seem similar to Algorithm 2 and 3 introduced in Chapter 2. However, as discussed before because of the specialty of linear convolution, the design of the matrix P should be different. We do not give the form of P here and will leave the design together with the proof of QIT security to the security analysis in Chapter 5.

Multiple stages of linear filtering are often used in image processing such as separable filtering (horizontal and vertical filtering) or wavelet transform (multiple stages of subband filtering). One advantage of our designed protocol is that it can be directly applied to multiple stage linear filtering.

4.1.3 QIT Three-party Solution

In this part, we will show how to implement the secure linear iflteirng with the help of a third party Clark, who we assume will not collude with either Bob or Alice. With the help of a third party, the protocol for linear filtering can be made *Information Theoretically Secure*. On the other hand, however, its application is comparatively limited as a

non-colluding third party may not be always present. The basic idea is that, instead of using matrix transforms, we randomly inject random noise into the rows of X_w and $\mathbf{h}$ for each inner product operation. The dependency between successive rows vanishes as random noise is used each time.

The proposed protocol is shown Algorithm 7, 8, and 9. The problem notation is the same as in Section 4.1.1 where Alice holds X_w and Bob holds $\mathbf{h}$ and they want to jointly compute $X_w\mathbf{h}$. Alice generates a random $n \times m$-dimensional matrix X_a and computes $X_b = X_w - X_a$. Similarly, Bob generate a random m-dimensional vector $\mathbf{h}_a$ and compute $\mathbf{h}_b = \mathbf{h} - \mathbf{h}_a$. Then the inner product can be rewritten as

$$X_w\mathbf{h} = X_a\mathbf{h}_a + X_a\mathbf{h}_b + X_b\mathbf{h}_a + X_b\mathbf{h}_b, \tag{4.2}$$

Note X_a or X_b alone provides no information about X_w as proved in Chapter 5. Neither does $\mathbf{h}_a$ or $\mathbf{h}_b$ alone about $\mathbf{h}$. Unfortunately, it is impossible for Bob and Alice to compute all the four items in Equation 4.2 by just receiving one component of the vector from each other. For example, if Alice sends Bob X_a, he can computer the first and second terms $\mathbf{r}_b = X_a\mathbf{h}_a + X_a\mathbf{h}_b$. If then Bob send Alice $\mathbf{h}_a$, Alice can then compute the fourth but not the third. To solve this conundrum, we introduce a third party Clark. If Bob sends Alice $\mathbf{h}_a$ an Alice computes $\mathbf{r}_a = X_b\mathbf{h}_a$, Alice and Bob can send Clark X_b and $\mathbf{h}_b$ so that Clark can compute the remaining term in Equation

4.2, i.e. $\mathbf{r}_c = X_b\mathbf{h}_b$. Provided that no two parties collude with each other, the information Alice, Bob and Clark have are all random data which disclose no information about either X_w and $\mathbf{h}$. Therefore, the protocol does achieve *Information Theoretic Security* (Since the proof is obvious, it is omitted in Chapter 5.).

Algorithm 7 `3PartyInnerProductAlice`(X_w)

Require: $X_w \in \mathbb{R}^{n\times m}$, which is reformated from the original image.

1: Generate random matrix X_a.
2: $X_b \leftarrow X_w - X_a$.
3: Transmit X_a to Bob.
4: Transmit X_b to Clark.
5: Receive $\mathbf{h}_a$ from Bob.
6: **return** $\mathbf{r}_a = X_b\mathbf{h}_a$

Algorithm 8 `3PartyInnerProductBob`($\mathbf{h}$)

Require: $\mathbf{h} \in \mathbb{R}^m$.

1: Generate random matrix $\mathbf{h}_a$.
2: $\mathbf{h}_b \leftarrow \mathbf{h}_w - \mathbf{h}_a$.
3: Transmit $\mathbf{h}_a$ to Alice.
4: Transmit $\mathbf{h}_b$ to Clark.
5: Receive X_a from Bob.
6: **return** $\mathbf{r}_b = X_a\mathbf{h}_a + X_a\mathbf{h}_b$

Algorithm 9 `3PartyInnerProductClark`()

1: Receive X_b from Alice.
2: Receive $\mathbf{h}_b$ from Bob.
3: **return** $\mathbf{r}_a = X_b\mathbf{h}_b$

At the end of the protocol, Alice, Bob and Clark will have $\mathbf{r}_a$, $\mathbf{r}_b$, and $\mathbf{r}_c$ respectively such that the output image is $y = \mathbf{r}_a + \mathbf{r}_b + \mathbf{r}_c$. The correctness of our protocol can be easily seen from Equation 4.2. To perform a second stage filtering with, say Bob' $\mathbf{g}$, Clark can first generate a random vector $\mathbf{r}_{c1}$, and send it to Alice, while on the other

hand, send $\mathbf{r}_{c2} = \mathbf{r}_c - \mathbf{r}_{c1}$ to Bob. Alice and Bob add the received vector to the quantity they already hold to have $\mathbf{r}'_a = \mathbf{r}_r + \mathbf{c}_{c1}$ and $\mathbf{r}'_b = \mathbf{r}_b + \mathbf{r}_{c2}$ respectively. Then we can simply apply the distribution rule for convolution $y \otimes \mathbf{g} = \mathbf{r}'_a \otimes \mathbf{g} + \mathbf{r}'_b \otimes \mathbf{g}$. Since Bob knows $\mathbf{r}'_b$ and $\mathbf{g}$, he can computer $\mathbf{r}'_b \otimes \mathbf{g}$ himself. $\mathbf{r}'_a \otimes \mathbf{g}$ can then be computed using the three-party linear filtering protocol Among Alice, Bob, and Clark.

4.2 Thresholding

4.2.1 Classical Two-Party Solution

Research on secure thresholding problem in the cryptographic society gave it a different name, called *Secure Millionaire* problem, though essentially the same problem. Computationally secure protocols solving this problem was done by utilizing the concept of Oblivious Transfer. The original solution to this problem is given by Andrew Yao [37] in 1982. Shai Avidan used this protocol as part of his blind face detection algorithm in [4].

Alice and Bob each hold a secrete number a and b respectively and they want to compare who has a larger number. The classical solution utilizing the OT primitive is to first have Alice and Bob individually represent their number in binary format. The two numbers are then checked through OT bit by bit from the highest significant bit (HSB) to the lowest significant bit (LSB). Both parties will not know the final answer until the LSB has been checked. For each bit, Bob prepares a

look-up table based on his current bit and all the two possible values of Alice's bit. The details are shown in Algorithm 10 [4].

Algorithm 10 `SecureMillionaire`(a, b)

Require: $a, b \in F$, where F is some finite field. Suppose m-bit is long enough to represent a and b, and let $a_i,\ b_i$ be the i-th bit of $a,\ b$.

1: Bob defines three states $\{\mathcal{A}, \mathcal{B}, \mathcal{C}\}$, which corresponding to $\mathcal{A}$lice's number is larger, $\mathcal{B}$ob's number is larger, and the relationship is $\mathcal{U}$ndecided respectively. For each round of communication, Bob encrypt the three states using a random permutation of the numbers $\{1, 2, 3\}$.

2: For the Bob's HSB b_m, He constructs a 2-entry table from the following lookup table.

	$b_m = 0$	$b_m = 1$
$a_m = 0$	$\mathcal{U}$	$\mathcal{B}$
$a_m = 1$	$\mathcal{A}$	$\mathcal{U}$

The lookup table is built according to Bob's possible b_m and Alice's possible a_m. If $b_m = 0$ Bob should extract the left column as the 2-entry table, otherwise, Bob should use the right column.

3: Alice communicates with Bob through OT_1^2 to obtain the state s_n according to her a_m.

4: **for** $i \leftarrow m - 1$ to 1 **do**

5: Bob construct a 6-entry table from the following lookup table which is indexed by s_{i+1} and a_i.

	$b_i = 0$	$b_i = 1$
$s_{i+1} = \mathcal{A} \wedge a_i = 0$	$\mathcal{A}$	$\mathcal{A}$
$s_{i+1} = \mathcal{B} \wedge a_i = 0$	$\mathcal{B}$	$\mathcal{B}$
$s_{i+1} = \mathcal{U} \wedge a_i = 0$	$\mathcal{U}$	$\mathcal{B}$
$s_{i+1} = \mathcal{A} \wedge a_i = 1$	$\mathcal{A}$	$\mathcal{A}$
$s_{i+1} = \mathcal{B} \wedge a_i = 1$	$\mathcal{B}$	$\mathcal{B}$
$s_{i+1} = \mathcal{U} \wedge a_i = 1$	$\mathcal{U}$	$\mathcal{U}$

where s_{i+1} is the stata obtained from previous round of communication. If Bob's $a_i = 0$ he should use the left column as the 6-entry table, otherwise he should use the right column.

6: Alice communicates with Bob through OT_1^6 with the combination of s_{i+1} and a_i as her index to obtain s_i from the table.

7: **end for**

8: Bob send Alice the meaning of the three states of s_1 corresponding to the LSB and Alice knows which number is larger.

9: If she wants, Alice can send the final result to Bob.

10: **return** $\mathcal{A}$lice win, $\mathcal{B}$ob win or $\mathcal{E}$qual.

Note it is quite possible that in some intermediate bit, the relation-

ship between a and b can be decided. However, if the protocol stops right after the relationship is decided, given the round number, it will leave much information about the opposite party's number since both of them knows the number they have. For example, if the relationship is decided at the first round and Bob's $b_m = 1$, then he knows that $a_m = 0$ for sure and Alice's number cannot be larger than 2^m, which is much different if at the last round Bob knows that he has a larger number, then any number less than Bob's b is possible for Alice's a.

On the other hand, to prevent Alice from interpreting the meaning of the states $\{\mathcal{A}, \mathcal{B}, \mathcal{U}\}$, each round Bob should encrypt the three state with a regenerated random permutation of the numbers $\{1, 2, 3\}$. For example, if for the 1^{st} round, Bob use 1 to represent the state $\mathcal{A}$, at the 2^{nd} round after regeneration of the random permutation, he could use 3 to represent the state $\mathcal{A}$. Thus, as each round even after Alice received a number from the set $\{1, 2, 3\}$, she won't be able to know which state the number represents.

4.2.2 QIT Two-Party Solution

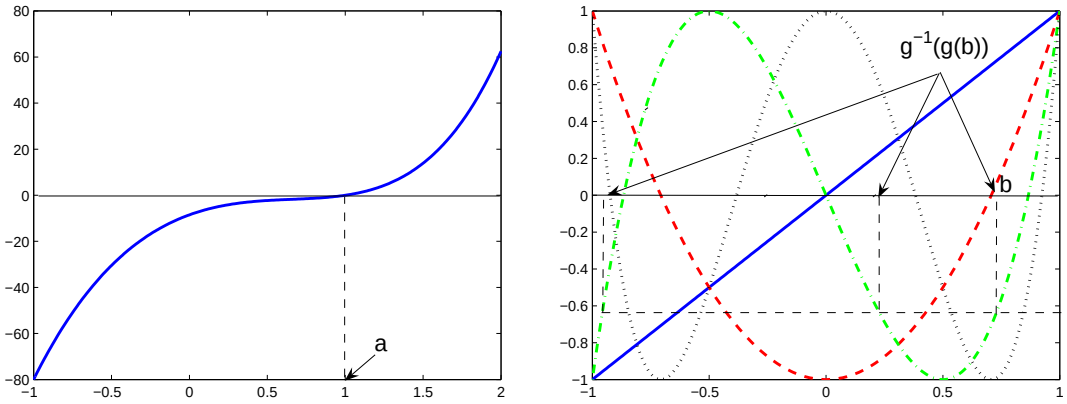
Assume we have two distrusted parties: Alice and Bob. Alice holds a secret scaler a, and Bob holds another secret scaler b. They want to find out who has a bigger number without disclosing their private data. Under our new notion of security, we propose to convert this problem into a special polynomial evaluation problem. Let n be an even num-

ber. Alice first randomly generates a $(n-1)^{\text{th}}$-degree polynomial $f(x)$ that has only one real root: Alice's secret number a. In addition, we require that the derivative of $f(x)$ at a is non-negative. Alice can easily generate this polynomial by first randomly selecting $(n-2)/2$ complex conjugate numbers as the roots of the polynomial, and then multiplying the resulting polynomial by a negative random number if the derivative of f at a is negative or a positive random number otherwise. We will refine this procedure for better security in Chapter 5. The key property of $f(x)$ is that for any $b > a$, we have $f(b) > 0$ and for all $b < a$, we have $f(b) < 0$. An example of such a $f(x)$ is shown in Figure 4.1(a). Thus if Bob knows only the value of $f(b)$ without knowing the actual polynomial, he can easily solve the problem without any knowledge of a. Given $f(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$, we can evaluate $f(b)$ as an inner product between two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$f(b) \triangleq a_{n-1}b^{n-1} + \cdots + a_1 b + a_0 \triangleq \mathbf{x}_1^T \mathbf{x}_2 \tag{4.3}$$

where Alice has $\mathbf{x}_1 = [\, a_{n-1} \ \cdots \ a_1 \ a_0 \,]^T$ and Bob has $\mathbf{x}_2 = [\, b^{n-1} \ \cdots \ b \ 1 \,]^T$.

Thus, the evaluation of a polynomial becomes that of an inner product. Our secure inner product evaluation is based on [11]. The idea is to linearly map $\mathbf{x}_1$ and $\mathbf{x}_2$ into a lower-dimensional space such that given the transformed results, it is impossible to exactly recover a and b. We use an invertible matrix $M \in R^{n \times n}$, and vertically divide it



(a) (b)

Figure 4.1: *(a) Random polynomial with a single real root at $a = 1$. (b) Chebyshev's polynomials of degree one (blue solid), degree two (red dash), degree three (green dast-dot) and degree four (black dot).*

into two parts $M_l \in R^{n\times k}$ and $M_r \in R^{n\times(n-k)}$. On the other hand, we horizontally divide M^{-1} into two parts $M_t \in R^{k\times n}$ and $M_b \in R^{(n-k)\times n}$.

Well the readers may have noticed that we basicaly use the same idea of matrix transformation for linear filtering and thresholding. However, the notation of the transformation matrices are different. The reason we use different notations is because the design of these matrices are different for different problems. Hence, different notations are used for better correspondences to Chapter 5 when we analyze the security of the proposed protocols.

The design of M and its submatrices is critical to the security of the protocol and the details will be discussed in Chapter 5. Given M and the submatrices, our protocol of secure thresholding is described in Algorithm 11 and 12.

Algorithm 11 `ThresholdingAlice`$(\mathbf{x}_1, M)$

Require: $\mathbf{x}_1 = [\ a_{n-1}\ \cdots\ a_1\ a_0\ ]^T \in \mathbb{R}^n$. $M = (M_l\ \ M_r)$ is a $n \times n$ invertible matrix where $n \geq 2$; $M_l \in \mathbb{R}^{n \times k}$ and $M_r \in \mathbb{R}^{n \times (n-k)}$.

1: $\mathbf{x}_{11} \leftarrow \mathbf{x}_1^T M_l$
2: $\mathbf{x}_{12} \leftarrow \mathbf{x}_1^T M_r$
3: Transmit $\mathbf{x}_{12}$ to Bob.
4: Receive $\mathbf{x}_{21}$ from Bob.
5: Send $\mathbf{x}_{11}^T \mathbf{x}_{21}$ to Bob.

Algorithm 12 `ThresholdingBob`$(\mathbf{x}_2, M^{-1})$

Require: $\mathbf{c} = [\ b^{n-1}\ \cdots\ b\ 1\ ]^T \in \mathbb{R}^n$. $M^{-1} = \begin{pmatrix} M_t \\ M_b \end{pmatrix}$ is a $n \times n$ invertible matrix where $n \geq 2$; $M_t \in \mathbb{R}^{k \times n}$ and $M_b \in \mathbb{R}^{(n-k) \times n}$.

1: $\mathbf{x}_{21} \leftarrow M_t \mathbf{x}_2$
2: $\mathbf{x}_{22} \leftarrow M_b \mathbf{x}_2$
3: Transmit $\mathbf{x}_{21}$ to Alice.
4: Receive $\mathbf{x}_{12}$ from Alice.
5: Receive $\mathbf{x}_{11}^T \mathbf{x}_{21}$ from Alice.
6: Compute $f(b) = \mathbf{x}_{12}^T \mathbf{x}_{22} + \mathbf{x}_{11}^T \mathbf{x}_{21}$
7: Return $f(b) > 0$.

The correctness of this protocol can be easily verified.

$$\begin{aligned} f(b) &= \mathbf{x}_1^T \mathbf{x}_2 \\ &= \mathbf{x}_1^T M M^{-1} \mathbf{x}_2 \\ &= \mathbf{x}_1^T \Big(M_l\ \ M_r \Big) \begin{pmatrix} M_t \\ M_b \end{pmatrix} \mathbf{x}_2 \\ &= \mathbf{x}_{11}^T \mathbf{x}_{21} + \mathbf{x}_{12}^T \mathbf{x}_{22} \end{aligned}$$

For a three-party case, given the non-colluding third party Clark, the solution to this problem becomes obvious. Alice and Bob can just send their numbers to Clark, and he compares the two number and tell them who has a larger number.

Chapter 5

Security Analysis

In this section, we will show that our proposed protocols (linear filtering and thresholding) is QIT secure.

5.1 Linera Filtering Protocol

Under our assumption of semi-honest parties, the security of the protocol depends solely on how much information Alice and Bob can learn from the data they receive during the process of the protocol. Let's review Algorithm 5 and 6. Alice received $\mathbf{h}_1 = T\mathbf{h}$ from Bob, and Bob received $X_2 = X_w R$ from Alice. To satisfy our QIT security model, by Definition 3 and Definition 4, it is enough to show that $\forall X_w \in R^{n\times m}$, $\exists X'_w \in R^{n\times m}$, where X_w and X'_w are QIT indistinguishable under the mapping function R, which is true iff R are *noninvertable*. Yet on the other hand, we need also to have T be to *noninvertible* to make the protocol QIT secure. The property of T to be rank deficient, however, makes the statement automatically true, i.e. it is always QIT secure for Bob.

To make the protocol QIT secure for Alice, we need to consider the essense of the problem, linear convolution. Remember X_w is constructed by sliding a window (size of the filter) across the image to form rows of the matrix. Hence, $X_w \in \mathbb{R}^{n\times m}$ cannot span the whole space of $\mathbb{R}^{n\times m}$ because of the overlapping between adjacent rows. To simplify the problem, instead of a 2-D linear convolution, we will first discuss a 1-D linear convolution.

For any 1-D discrete signal $x(u)$, and a given filter $h(v)$, let the matrix after reformating $x(u)$ be $X_u \in \mathbb{R}^{n\times m1}$ and the vector form of $h(v)$ be $\mathbf{h}_v \in \mathbb{R}^{m\times 1}$. Then, the 1-D linear convolution can be written into a matrix product form as

$$\mathbf{y} = X_u \mathbf{h}_v. \tag{5.1}$$

Then, for the X_u formed by 1-D discrete signal $x(u)$, we have the following theorem.

Theorem 1. *Let $\gamma_1, \gamma_2, \cdots, \gamma_d$ be d random numbers, and*

$$L = \text{span}\left(\begin{bmatrix} 1 \\ \gamma_1 \\ \vdots \\ \gamma_1^{m-1} \end{bmatrix} \begin{bmatrix} 1 \\ \gamma_2 \\ \vdots \\ \gamma_2^{m-1} \end{bmatrix} \cdots \begin{bmatrix} 1 \\ \gamma_{\lfloor \frac{m}{2} \rfloor} \\ \vdots \\ \gamma_{\lfloor \frac{m}{2} \rfloor}^{m-1} \end{bmatrix}\right) \in \mathbb{R}^{m\times d}.$$

Let $x(u)$ be any 1-D discrete signal and X_u be the matrix reformatted from $x(u)$ as in Equation 5.1. If $R \perp L$, then $f(X_u) = X_u R$ is n*oninvertible for X_u.*

[1]Boundary handling is not a concern of us.

Proof Let $X_u(\mu,:)$ be the μ^{th} row of X_u.

$$X_u(\mu,:) = \begin{bmatrix} x(\mu) \\ x(\mu+1) \\ \vdots \\ x(\mu+m-1) \end{bmatrix}^T. \tag{5.2}$$

Since

$$L = \text{span}\left(\begin{bmatrix} 1 \\ \gamma_1 \\ \vdots \\ \gamma_1^{m-1} \end{bmatrix} \begin{bmatrix} 1 \\ \gamma_2 \\ \vdots \\ \gamma_2^{m-1} \end{bmatrix} \cdots \begin{bmatrix} 1 \\ \gamma_{\lfloor \frac{m}{2} \rfloor} \\ \vdots \\ \gamma_{\lfloor \frac{m}{2} \rfloor}^{m-1} \end{bmatrix} \right) \in \mathbb{R}^{m \times d}.$$

and $R \perp L$, the vectors $[1 \ \gamma_i \ \cdots \ \gamma_i^{m-1}],\ 1 \leq i \leq d$ are the left null space vectors of R. Then we have

$$X_u(\mu,:)R = X_u(\mu,:)R + \beta_1\gamma_1^\mu \begin{bmatrix} 1 \\ \gamma_1^1 \\ \vdots \\ \gamma_1^{m-1} \end{bmatrix}^T R + \cdots + \beta_d\gamma_d^\mu \begin{bmatrix} 1 \\ \gamma_d^1 \\ \vdots \\ \gamma_d^{m-1} \end{bmatrix}^T R. \tag{5.3}$$

Hence, if X'_u is the matrix reformatted by $x'(u) = x(u) + \beta_1\gamma_1^u + \cdots + \beta_d\gamma_d^u$, from Equation (5.3), we know that

$$X_u R = X'_u R. \tag{5.4}$$

As a result, $x'(u)$ is in the *QIT indistinguishable* set of $x(u)$ under $f(X_u) = X_u R$. Q.E.D.

Furthermore, we have the following corollary.

Corollary 1. *For 1-D discrete signal $x(u)$, If $Q = (L\ R)$, where L and R are as constructed in Theorem 1, such that Q has an inverse*

and is denoted as Q^{-1}, then Algorithm 5 and 6 are QIT secure under the transformation matrix $P = Q^{-1}$.

It is easy to see the correctness of COROLLARY 1. Hence, the proof is omitted here.

For a 2-D image, the method to construct such P and P^{-1} is very much similar to that in THEOREM 1. Therefore, we write it as another corollary of THEOREM 1.

COROLLARY 2. *Let $x(\mu, \nu)$ and X_w be the image and corresponding reformatted matrix, and $\mathbf{h}$ be the reformatted filter vector. The $(iN + j)^{\text{th}}$ row of X_w is reformatted from the window*

$$\begin{bmatrix} x(i,j) & x(i,j+1) & \cdots & x(i,j+l_2) \\ x(i+1,j) & x(i+1,j+1) & \cdots & x(i+1,j+l_2) \\ \vdots & \vdots & \ddots & \vdots \\ x(i+l_1,j) & x(i+l_1,j+1) & \cdots & x(i+l_1,j+l_2) \end{bmatrix},$$

such that

$$X_w(iN+j,:) = \begin{bmatrix} x(i,j) \\ \vdots \\ x(i,j+l_2) \\ \vdots \\ \vdots \\ \vdots \\ x(i+l_1,j) \\ \vdots \\ x(i+l_1,j+l_2) \end{bmatrix} \tag{5.5}$$

Let $\gamma_1, \cdots, \gamma_{d_1}$ *and* $\eta_1, \cdots, \eta_{d_2}$ *be random numbers, and*

$$L = \text{span}\left(\begin{bmatrix} 1 \\ \gamma_1\eta_1 \\ \vdots \\ \gamma_1\eta_1^{l_2} \\ \vdots \\ \vdots \\ \gamma_1^k \\ \gamma_1^k\eta_1 \\ \vdots \\ \gamma_1^k\eta_1^{l_2} \\ \vdots \\ \vdots \\ \gamma_1^{l_1} \\ \gamma_1^{l_1}\eta_1 \\ \vdots \\ \gamma_1^{l_1}\eta_1^{l_2} \end{bmatrix} \cdots \begin{bmatrix} 1 \\ \gamma_i\eta_j \\ \vdots \\ \gamma_i\eta_j^{l_2} \\ \vdots \\ \vdots \\ \gamma_i^k \\ \gamma_i^k\eta_j \\ \vdots \\ \gamma_i^k\eta_j^{l_2} \\ \vdots \\ \vdots \\ \gamma_i^{l_1} \\ \gamma_i^{l_1}\eta_j \\ \vdots \\ \gamma_i^{l_1}\eta_j^{l_2} \end{bmatrix} \cdots \begin{bmatrix} 1 \\ \gamma_{d_1}\eta_{d_2} \\ \vdots \\ \gamma_{d_1}\eta_{d_2}^{l_2} \\ \vdots \\ \vdots \\ \gamma_{d_1}^k \\ \gamma_{d_1}^k\eta_{d_2} \\ \vdots \\ \gamma_{d_1}^k\eta_{d_2}^{l_2} \\ \vdots \\ \vdots \\ \gamma_{d_1}^{l_1} \\ \gamma_{d_1}^{l_1}\eta_{d_2} \\ \vdots \\ \gamma_{d_1}^{l_1}\eta_{d_2}^{l_2} \end{bmatrix} \right) \in \mathbb{R}^{(l_1+1)(l_2+1)\times d_1 d_2} \tag{5.6}$$

Then if $Q = (L\ R)$*, where* $R \perp L$*, has an inverse and is denoted as* Q^{-1}*, then Algorithm 5 and 6 is QIT secure under the transformation matrix* $P = Q^{-1}$*.*

Proof By the same rationale as in THEOREM 1, let

$$x'(\mu,\nu) = x(\mu,\nu) + \beta_{11}\gamma_1^\mu\eta_1^\nu + \cdots + \beta_{ij}\gamma_i^\mu\eta_j^\nu + \cdots + \beta_{d_1d_2}\gamma_{d_1}^\mu\eta_{d_2}^\nu,$$

and the corresponding reformatted matrix be X'_w. Then, we know that

$$X_w R = X'_w R. \tag{5.7}$$

Hence, $x'(\mu,\nu)$ is in the QIT indistinguishable set of $x(\mu,\nu)$ under the

mapping $f(X_w) = X_w R$, such that the algorithm is QIT secure to Alice. Since the mapping for $\mathbf{h}$ automatically satisfies the QIT model, we conclude that the algorithm is QIT secure to both parties. Q.E.D.

5.2 Threholding Procotol

To analyze how secure our thresholding protocol is, we need to find out how much Alice and Bob can know from the data they send to each other. First, let us consider the information Bob sent to Alice. Bob sends Alice $\mathbf{x}_{21} = M_t \mathbf{x}_2$. Since M_t is a $k \times n$ matrix and $\mathbf{x}_2 = [\ b^{n-1}\ \cdots\ b\ 1\ ]^T$, $M_t \mathbf{x}_2$ is equivalent to evaluating k different polynomials at b, whose coefficients are defined by the row vectors of M_t. The cryptosystem induced by M_t is m-QIT secure if and only if there are at least m distinct values in the QIT indistinguishable set of b. This is equivalent to saying that the $(n-1)^{th}$ degree polynomials with coefficients $[M_t(i,1)\ M_t(i,2)\ \ldots\ M_t(i,n-1)\ M_t(i,n) - \mathbf{x}(i)]$ for $i = 1, 2, \ldots, k$ share m distinct roots. To maximize the security, we would to have m as large as $n-1$ which is the degree of the polynomials. As shown below, this constraint impose a maximum value on k, the number of rows in M_t, one can use. To show this, let us start from the following lemma:

Lemma 1. *Given two polynomials $g(x)$ and $h(x)$ of degree $n-1$ and a scalar b. If equations $g(x) = g(b)$ and $h(x) = h(b)$ share exactly the same roots, then $g(x) = k_1 h(x) + k_2$, where $k_1 \neq 0$ and k_2 are constants.*

Proof Since $g(x) = g(a)$ and $h(x) = h(a)$ share the same set of roots, we have $[g(x)-g(a)] = k_1[h(x)-h(a)]$ or $g(x) = k_1h(x)+[g(a)-k_1h(a)]$ for some $k_1 \neq 0$. Set $k_2 = g(a) - k_1h(a)$ and results follow. Notice that as long as $g(x)$ is not a constant, the coefficient vector of $f(x)$ is linear independent of the coefficient vector of $g(x)$. Q.E.D.

Theorem 2. *If the proposed thresholding protocol is* $(n-1)$*-QIT secure with respect to Bob, then the number of rows* k *in* M_t *is at most two.*

Proof Since the full matrix M^{-1} invertible, the k row vectors of M_t must be linearly independent. k is at least two based on Lemma 1. If k is larger than two, select any three row vectors and formulate the three corresponding polynomials $f_1(x)$, $f_2(x)$ and $f_3(x)$. Using Lemma 1, we have $f_1(x) = k_0f_3(x) + k_1$ and $f_2(x) = k_3f_3(x) + k_4$. Thus, the coefficient vectors of both $f_1(x)$ and $f_2(x)$ lie in the subspace spanned by the coefficient vector of $f_3(x)$ and $[\ 0\ \cdots\ 0\ 1\ ]^T$ and we obtain a contradiction. Q.E.D.

Next, we come to the actual design of M_t. Even though Alice may not know the precise value of b, she can usually assume b to be within a certain range. Without loss of generality, assume that $b \in [-1, 1]$. Thus, we need to find a polynomial $g(x)$ such that for any $b \in [-1, 1]$, all the $n - 1$ roots of $g(x) = g(b)$ are real and fall within the range $[-1, 1]$. An example of such function is the $(n-1)^{\text{th}}$ order Chebyshev's

polynomial[2]: $T_{n-1}(x) = \cos[(n-1)\cos^{-1}(x)]$. Figure 4.1(b) shows the first four Chebyshev's polynomials. We state the following fact without proof about the Chebyshev's polynomials though it is QITe obvious based on the figure.

Fact 1. *Except for at most $n+1$ distinct points within $[-1,1]$, the n^{th} order Chebyshev's polynomial $T_n(x)$ is n-noninvertible on $[-1,1]$*

The $n+1$ distinct points forms a measure-zero set in $[-1,1]$. Thus, the mapping $M_t\mathbf{x}_2$ will be $(n-1)$-QIT secure to Bob if we can set $M_t = \begin{pmatrix} C[T_{n-1}(x)] \\ C[k_0 T_{n-1}(x) + k_1] \end{pmatrix}$ where the operator $C[\cdot]$ denotes the coefficient vector of a polynomial. Given M_t, we can easily compute M_b by extending the two row vectors in M_t to a full set of basis in R^n.

We now show that the proposed thresholding protocol is also QIT-secure to Alice. Bob receives $\mathbf{x}_{12} = \mathbf{x}_1^T M_r$ from Alice. Bob also knows that $\mathbf{x}_1$ corresponds to the coefficient vector of a $(n-1)^{\text{th}}$ degree polynomial $f(x)$ with a single real root and non-negative derivative at that root. To show that the protocol is QIT-secure to Alice, we need to find $\mathbf{x}_1'$ that corresponds to a polynomial with the same features and $\mathbf{x}_{12} = \mathbf{x}_1'^T M_r$. Given M_t is defined based on the Chebyshev's polynomials, we have the following theorem:

Theorem 3. *Given that $\mathbf{x}_1$ is the coefficient vector of a polynomial $f(x)$ with only a single real root and non-negative derivative at that*

[2]Though stated in its general form, Chebyshev's polynomials can be easily computed as a true polynomial based on the recurrence relation $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$ with $T_{-1}(x) = 0$ and $T_0(x) = 1$.

root and $M_l = \begin{pmatrix} \mathbf{t}_1^T \\ \mathbf{t}_2^T \end{pmatrix}$, *there exists* $\mathbf{x}'_1 \neq \mathbf{x}_1$ *such that* $\mathbf{x}_1'^T M_r = \mathbf{x}_1^T M_r$ *and* $\mathbf{x}'_1$ *corresponds to the coefficients of* $f'(x)$ *which also has a single real root with non-negative derivative at that root.*

Proof Recall that $M^{-1} = \begin{pmatrix} M_t \\ M_b \end{pmatrix}$ and $M = (M_l\ M_r)$. Thus, M_t and M_r relate to each other by the following relationship:

$$M_t \cdot M_r = 0$$

As M_r and M_t are a part of an invertible matrix, the rank of M_t is 2 and the rank of M_r is $n-2$. Thus, if $\mathbf{v}^T M_r = 0$, $\mathbf{v}^T$ must be in the subspace S spanned by the row vectors of M_t. Note that $(\mathbf{x}_1 + \mathbf{v})^T M_r = \mathbf{x}_1^T M_r$. Our strategy is to find an appropriate $\mathbf{v}$ that can satisfy the conditions.

On the other hand, the row vectors of M_t denote the coefficients of the Chebyshev's polynomial $T_{n-1}(x)$ and $k_0 T_{n-1}(x) + k_1$ for arbitrary k_0 and $k_1 \neq 0$. It is obvious that the vector $[\ 0\ \cdots\ 0\ 1\ ]^T$ is in the subspace S. Define $\mathbf{v} = [\ 0\ \cdots\ 0\ \epsilon/2\ ]^T$ where $-\epsilon$ is the *largest local maximum* in $(-\infty, a]$ of Alice polynomial $f(x)$. If no such local maximum exists, ϵ can be chosen arbitrarily. The vector $\mathbf{x}'_1 = \mathbf{x} + \mathbf{v}$ then corresponds to a polynomial $f'(x) = f(x) + \epsilon/2$. Note that this polynomial still has a single real root because the large local maximum on the left hand side of the root is still $\epsilon/2$ from zero. Furthermore, the derivative at the root must be non-negative otherwise a local maxima would have crossed the x-axis. Q.E.D.

In the unfortunate case when the largest local maximum left of a

and the smallest local minimum right of a are both small, we can only shift $f(x)$ by a small amount before it starts to have more than one real root. In other words, it is possible for Bob to roughly estimate a despite the fact that the protocol is QIT-secure. The security, however, can be significantly improved by imposing some constraints on the random complex roots of $f(x)$. Without loss of generality, we again assume that Alice's number $a \in [-1, 1]$. We have the following result:

Theorem 4. *The thresholding protocol is INFORMATION THEORETICALLY secure to Alice if Alice first generates an auxiliary polynomial*

$$g(x) = (x-1) \prod_{i=1}^{(n-2)/2} (x - c_i)(x - \bar{c_i}) \tag{5.8}$$

with random c_i under the constraint $\mathrm{Real}(c_i) > 1$ *for all i and then let* $f(x) = g(x) - g(a)$.

Proof For any real x, if we rewrite each term in Equation (5.8) in polar form, the complex exponential terms for the conjugate roots will cancel each other and $g(x)$ will become

$$g(x) = \mathrm{sign}(x-1) \cdot |x-1| \cdot \prod_{i=1}^{(n-2)/2} |x - c_i| \cdot |x - \bar{c_i}| \tag{5.9}$$

Equation (5.9) shows that a) $g(x)$ is negative for $x < 1$ and positive for $x > 1$ and b) $g(x)$ is strictly increasing or $\frac{dg}{dx} > 0$ for $x \leq 1$. This is because as the real parts of all the complex roots are larger than one, every modulus term in Equation (5.9) decreases as x approaches 1 from $-\infty$. As $\mathrm{sign}(x-1)$ is negative, $g(x)$ is strictly increasing.

Clearly $f(x) = g(x) - g(a)$ for $a \in [-1, 1]$ satisfies our requirements of having a single real root and non-negative derivative at a. Recall that the coefficient vector of $f'(x) = f(x) + c$ for any constant c is in the null space of M_r. By choosing $c \in [g(a), g(a) - g(-1)]$, $f'(x)$ can have its single real root anywhere in $[-1, 1]$. Thus, based on the information sent by Alice, Bob has no information about a and the protocol is information theoretically secure to Alice. Q.E.D.

In closing, we have developed a linear filtering protocol that is QIT security to both Alice and Bob, and a thresholding protocol that achieves perfect security for Alice but leaks some information about Bob's secret number (only QIT secure).

Chapter 6

Experiments and Discussion

In this chapter, we will show the experimental results and discuss some possible problems of our proposed protocols.

6.1 Experimental Results

In this section, comparison of the time used between our proposed protocols and the classic protocols will be presented. As will be seen, our proposed protocols speed up the computation significantly.

6.1.1 Linear Filtering

Our proposed linear filtering protocol is computationally efficient as expected compared with the classical OT based protocols. As a comparison, we have implemented a classic two-party protocol based on the decription from [4], using our own 512-bit RSA public-key cryptosystem (PKCS). We then compare its performance with the algorithm described in Chapter 4 on a dual Wintel CPU (P4-3.4GHz) desktop with 1GB memory. The reason we did not test the classic protocol on real

images is because it will take hours to do a linear filtering on a single image. The oblivious transfer based technique takes about 20 *minutes* to compute the inner product of two 20-dimensional vectors while our two-party protocol uses only 30 *milliseconds* and our three-party protocol uses 47 *milliseconds*. Despite our non-optimal implementation of the oblivious transfer protocol, its slow performance can be attributed to the handling of very long integers in the encryption/decryption process as well as the large amount of information exchanged between Alice and Bob. For linear filtering using a 7×7 Gaussian mask on the same computing platform, our two-party solution takes on average 0.7 *seconds* to denoise a 128×128 image and our three party solution takes around 0.6 *seconds*. We summarize the timing in Table 6.1.

Table 6.1: Average time used for linear filtering.

	OT Based	Two-party	Three-party
Inner Product of 20-D vectors	20 minutes	30 milliseconds	47 milliseconds
Image Linear Filtering	N/A	0.7 seconds	0.6 seconds

6.1.2 Thresholding

To compare the computational performance of the proposed thresholding protocol with existing schemes, we use the cryptographic secure millionaire protocol described in [4]. We have implemented both protocols in Matlab 7.0.1 on a Pentium 4 Dual Core 3.4GHz machine with 1GB memory. To ensure the validity of the protocols, the protocols for Bob and Alice are run separately in two processes and the two

protocols exchange information using TCP/IP.

For the cryptographic protocol, Bob creates a series of tables by bitwise comparing his secret number b with every possible value of Alice's secret number a, encrypts the tables using a public-key cipher, and then transfers them to Alice. Alice decrypts the only entry of the table that is corresponding to his own number a and extracts the results. We have implemented our own 512-bit RSA public-key cipher using the long-integer operations provided by the Maple kernel within Matlab. We have run a series of comparison between random pairs of 64-bit floating point numbers. The average computation time per pair on Bob's side is 84.70 seconds. Excluding the time spent on network operations, this number reduces to 83.73 seconds. The computation times per pair for Alice are 10.72 seconds with networking and 10.43 without. Alice is faster because she does not need to generate large tables. We have pre-generated a set of random public keys used in the protocol and have excluded the time for key generation in the measurement.

On the other hand, our proposed technique runs *significantly* faster. On average, Alice takes 35.40 milliseconds with network and 1.31 milliseconds without for each comparison. Bob takes 35.41 milliseconds with network and 0.23 milliseconds without. Alice takes longer as she needs to generate a 19^{th} order random polynomial. Compared with the cryptographic protocols, this is a factor of 10^4 improvement in com-

putation time. In summary, we listed the timing of both protocols in Table 6.2.

Table 6.2: Average time used for thresholding.

	Time Used
OT Based with Network	84.70 seconds
OT Based without Network	83.73 seconds
QIT with Network	35.40 milliseconds
QIT without Network	1.31 milliseconds

6.2 Discussion

Although our proposed protocols improve the computational time significantly, there are still points that need to be further investigated. One problem with our two-party linear filtering protocol is on the discontinuities of the images (edges). Since as can be seen in Chapter 5, for $x(\mu, \nu)$ the QIT indistinguishable set is given by $x'(\mu, \nu) = x(\mu, \nu) + \beta_{11}\gamma_1^\mu\eta_1^\nu + \cdots + \beta_{ij}\gamma_i^\mu\eta_j^\nu + \cdots + \beta_{d_1d_2}\gamma_{d_1}^\mu\eta_{d_2}^\nu$. Notice that $\beta_{ij}\gamma_i^\mu\eta_j^\nu$ is continuous for $i,\ j$. Therefore, the discontinuity points is contributed soly by $x(\mu, \nu)$. This is a possible weak point of leaking edge information about the image, which sometimes is very important. One possible remedy of this problem is to use original inner product protocol in [11], but before applying it, Alice need to random permute the rows of X_w as is proposed in [19]. This, however, arises another problem as the security of this protocol is still kept unproven.

Another problem is about the proposed thresholding protocol. At the end of the protocol, Alice or Bob needs to send her/his share to the

opposite side. Definitely, Bob cannot send his share to Alice, because Alice generated the polynomial $f(x)$, and if given Bob's share, Alice will know the value of $f(b)$, then she will know what b is simply by solving the equation $f(x) = f(b)$. This will tell everything about Bob's b. Hence, the only possible way is to have Alice send her share to Bob, and let Bob know $f(b)$. We know that Alice's $f(x)$ is transformed by matrix M_t and M_b, then it is decomposed to $f(x) = f_b(x)+f_t(x)$, where $f_b(x)$ is the polynomial in the space spanned by M_b and $f_t(x)$ in the space spanned by M_t. Receiving $\mathbf{x}_{12} = \mathbf{x}_1^T M_r$ and $\mathbf{x}_{11}^T \mathbf{x}_{21} = \mathbf{x}_1^T M_l M_t \mathbf{x}_2$ from Alice, Bob can estimate $f_b(x)$ and $f_t(x)$ respectively. Thus, the perfect secrecy for Alice is compromised.

Chapter 7

Conclusion

In this thesis, we proposed a novel security model called Quasi Information Theoretic (QIT) model. Compared with the two existing classical cryptographic security models, namedly Information Theoretic Security and Computational Security, our proposed model provides less security than the former model in the information sense while enable us to develop protocols that are significantly faster than those under the latter model. Under the proposed QIT security model, protocols to solve two problems, linear filtering and thresholding, are developed. The rigorous analysis of the security of the protocols for both parties were also presented. The experimental results showed that our proposed protocols improved the computational time largely. While there are some potential insecure point in our proposed protocols as is discussed in Chapter 6, we need further improvement and analysis in the future. Other future work includes extending the QIT framework to more signal and image processing algorithms.

Bibliography

[1] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the kth ranked element. In *Proceedings of Advances in Cryptology - EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques*, pages 40–55, 2004. 15

[2] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *In Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques*, pages 119–135, 2001. 14

[3] R. Anderson. *Trusted Computing Frequently Asked Questions*. http://www.cl.cam.ac.uk/∼rja14/tcpa-faq.html, 1.1 edition, August 2003. 2

[4] S. Avidan and M. Butman. Blind vision. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3953 of *LNCS*, pages 1–13. Springer, 2006. 3, 15, 16, 31, 32, 48, 49

[5] D. Boneh, E. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In Joe Killian, editor, *Proceedings of Theory of Cryptography Conference 2005*, volume 3378 of *LNCS*, pages 325–342. Springer-Verlag, 2005. 15

[6] Christian Cachin, Jan Camenisch, Joe Kilian, and Joy Muller. One-round secure computation and secure autonomous mobile agents. In *Automata, Languages and Programming*, pages 512–523, 2000. 15

[7] Y.-C. Chang and C.-J. Lu. Oblivious polynomial evaluation and oblivious neural learning. *Theoretical Computer Science*, 341:39–54, 2005. 15

[8] C. Clifton and et al. Tools for privacy preserving distributed data mining. *proceeding of SIGKDD Explorations*, 4(2):1–7, Dec 2002. 11

[9] R. Cramer and et al. Efficient multi-party computations with dishonest majority. *Proceedings of Eurocrypt'99*, pages 311–326, 1999. 12

[10] C. Crepeau, J. van de Graaf, and A. Tapp. Committed oblivious transfer and private multi-party computations. *Advances in Cryptology: Proceedings of Crypto'95*, pages 110–123, 1995. 12

[11] W. Du, Y. Han, and S. Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. *proc. of*

the 4th SIAM Int'l Conf. on Data Mining, pages 222–233, 2004. 5, 15, 17, 18, 27, 34, 51

[12] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM 28*, pages 637–647, 1985. 14

[13] Matthias Fitzi, Martin Hirt, and Ueli Maurer. Trading correctness for privacy in unconditional multi-party computation. *Advances in Cryptology - Crypto'98*, pages 121–136, 1998. 12

[14] W. Gasarch. A survey on private information retrieval. *The Bulletin of the EATCS*, 82:72–107, 2004. 15

[15] O. Goldreich. *Foundations of Cryptography: Volume II Basic Applications*. Cambridge, 2004. 2, 3, 10, 20

[16] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental games. *proceeding of 19th ACM Symposium on Theory of Computing*, pages 218–229, 1987. 10

[17] S. Goldwasser and M. Bellare. *Lecture Notes on Cryptography*. Massachusetts Institue of Technology, 2001. 14

[18] Martin Hirt and Ueli Maurer. Complete characterization of adversaries tolerable in secure multi-party computation. *Proceedings of PODC*, pages 25–34, 1997. 12

[19] N. Hu, S. Cheung, and T. Nguyen. Secure image filtering. *Proceeding of IEEE International Conference on Image Processing (ICIP) 2006*, pages 1553–1556, 2006. 15, 51

[20] N. Hu and S.-C. Cheung. A new security model for secure thresholding. In *Proc. of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP 2007)*, http://vis.uky.edu/mialab/Publications for Secure Image Processing.html, April 2007. 15

[21] Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings, 20th Annual ACM Symposium on the Theory of Computation (STOC)*, pages 20–31, 1988. 15

[22] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1996. 3, 13

[23] Y. Lindell and B. Pinkas. Privacy preserving data mining. *Advances in Cryptography - Crypto2000, LNCS, 1880*, pages 36–53, 2000. 11

[24] Y. Lindell and B. Pinkas. Privacy preserving data mining. *Journal of Cryptology*, 15(3):177–206, 2002. 15

[25] B. Malin, E. Airoldi, S. Edoho-Eket, and Y. Li. Configurable security protocols for multi-party data analysis with malicious participants. *Proceedings of the 21st IEEE International Conference on Data Engineering*, pages 533–544, 2005. 12

[26] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. *Proc. 12th Ann. Symp. Discrete Algorithms*, pages 448–457, 2001. 15

[27] Moni Naor and Kobbi Nissim. Communication complexity and secure function evaluation. *Electronic Colloquium on Computational Complexity (ECCC)*, 8(062), 2001. 15

[28] Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 245–254, 1999. 14, 15

[29] B. Pinkas. Cryptographic techniques for privacy-preserving data mining. *processing of SIGKDD Explorations*, (2):12–19, 2002. 11

[30] M. O. Rabin. How to exchange secrets by oblivious transfer. *Tech. Memo TR-81, Aiken Computation Laboratory*, 1981. 14

[31] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(1):612–613, 1979. 13

[32] C. E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28:656–715, Oct 1949. 20

[33] L. Trevisan. Some applications of coding theory in computational complexity. *Quaderni di matematica*, 13:347–424, 2004. 15

[34] Trusted Computing Group. *TCG Specification Architecture Overview*, revision 1.2 edition, April 2004. 2

[35] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. *proceeding of the 8th ACM SIGKDD*, pages 639–644, 2002. 11

[36] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1:511–518, 2001. 16

[37] A. C. Yao. Protocols for secure computation. *Proceedings of 23rd IEEE Symposium on Foundations of Computer Science*, pages 160–164, 1982. 4, 10, 31

[38] A. C. Yao. How to generate and exchange secrets. *27th FOCS*, pages 162–167, 1986. 3

Vita

Personal Particulars

- Date and Place of Birth: Feb 17th, 1980 Liaoning, China

Education

- National University of Singapore
  Master of Engineering in Electrical & Computer Engineering

- Peking University
  Bachelor of Science in Electronics
  Bachelor of Economics

Publications

1. Nan Hu, S. Cheung, "A New Security Model For Secure Thresholding", in proc. Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP) 2007.

2. Nan Hu, S. Cheung, T. Nguyen, "Secure Image Filtering", in proc. IEEE Int'l Conf. Image Processing (ICIP) 2006, pp. 1553-1556, 2006.

3. Nan Hu, W. Huang, S. Ranganath, "Robust Attentive Behavior Detection by Non-Linear Head Pose Embedding and Estimation", in proc. 9th European Conf. Computer Vision (ECCV), vol. 3, pp. 356-367, 2006.

4. Nan Hu, W. Huang, S. Ranganath, "Attentive Behavior Detection by Non-Linear Head Pose Embedding and Mapping", in proc. IEEE 7th workshop Multimedia Signal Processing (MMSP), pp. 1-4, 2005.

5. Nan Hu, W. Huang, S. Ranganath, "Head Pose Estimation by Non-Linear Embedding and Mapping", in proc. IEEE Int'l Conf. Image Processing (ICIP) 2005, vol. 2, pp. 342-345, 2005.

6. Nan Hu, W. Huang, S. Ranganath, "Fast Detection of Frequent Change in Focus of Human Attention", in L. Paletta et al. (Eds.): WAPCV2004, LNCS3368, pp.216-230, Springer-Verlag, 2005.

Nan HU

(Typed Name of Student)