Université de Montréal

Key Agreement Against Quantum Adversaries

par
Kassem Kalach

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des arts et des sciences
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en informatique

Août, 2012

RÉSUMÉ

Un protocole d'échange de clés est un scénario cryptographique entre deux partis légitimes ayant besoin de se mettre d'accord sur une clé commune secrète via un canal public authentifié où tous les messages sont interceptés par un espion voulant connaître leur secret. Nous considérons un canal classique et mesurons la complexité de calcul en termes du nombre d'évaluations (requêtes) d'une fonction donnée par une boîte noire.

Ralph Merkle fut le premier à proposer un schéma non classifié permettant de réaliser des échanges securisés avec des canaux non securisés. Lorsque les partis légitimes sont capables de faire $O(N)$ requêtes pour un certain paramètre N, tout espion classique doit faire $\Omega(N^2)$ requêtes avant de pouvoir apprendre leur secret, ce qui est optimal. Cependant, un espion quantique peut briser ce schéma avec $O(N)$ requêtes. D'ailleurs, il a été conjecturé que tout protocole, dont les partis légitimes sont classiques, pourrait être brisé avec $O(N)$ requêtes quantiques.

Dans cette thèse, nous introduisons deux catégories des protocoles à la Merkle. Lorsque les partis légitimes sont restreints à l'utilisation des ordinateurs classiques, nous offrons le premier schéma classique sûr. Il oblige tout adversaire quantique à faire $\Omega(N^{13/12})$ requêtes avant d'apprendre le secret. Nous offrons aussi un protocole ayant une sécurité de $\Omega(N^{7/6})$ requêtes. En outre, pour tout $k \geqslant 2$, nous donnons un protocole classique pour lequel les partis légitimes établissent un secret avec $O(N)$ requêtes alors que la stratégie optimale d'espionnage quantique nécessite $\Theta\big(N^{\frac{1}{2}+\frac{k}{k+1}}\big)$ requêtes, se rapprochant de $\Theta(N^{3/2})$ lorsque k croît.

Lors les partis légitimes sont équipés d'ordinateurs quantiques, nous présentons deux protocoles supérieurs au meilleur schéma connu avant ce travail. En outre, pour tout $k \geqslant 2$, nous offrons un protocole quantique pour lequel les partis légitimes établissent un secret avec $O(N)$ requêtes alors que l'espionnage quantique optimale nécessite $\Theta\big(N^{1+\frac{k}{k+1}}\big)$ requêtes, se rapprochant de $\Theta(N^2)$ lorsque k croît.

Mots clés : Cryptographie, Algorithmes quantiques, Bornes inférieures quantiques, Oracle aléatoire.

ABSTRACT

Key agreement is a cryptographic scenario between two legitimate parties, who need to establish a common secret key over a public authenticated channel, and an eavesdropper who intercepts all their messages in order to learn the secret. We consider query complexity in which we count only the number of evaluations (queries) of a given black-box function, and classical communication channels.

Ralph Merkle provided the first unclassified scheme for secure communications over insecure channels. When legitimate parties are willing to ask $O(N)$ queries for some parameter N, any classical eavesdropper needs $\Omega(N^2)$ queries before being able to learn their secret, which is is optimal. However, a quantum eavesdropper can break this scheme in $O(N)$ queries. Furthermore, it was conjectured that any scheme, in which legitimate parties are classical, could be broken in $O(N)$ quantum queries.

In this thesis, we introduce protocols à la Merkle that fall into two categories. When legitimate parties are restricted to use classical computers, we offer the first secure classical scheme. It requires $\Omega(N^{13/12})$ queries of a quantum eavesdropper to learn the secret. We give another protocol having security of $\Omega(N^{7/6})$ queries. Furthermore, for any $k \geqslant 2$, we introduce a classical protocol in which legitimate parties establish a secret in $O(N)$ queries while the optimal quantum eavesdropping strategy requires $\Theta\big(N^{\frac{1}{2}+\frac{k}{k+1}}\big)$ queries, approaching $\Theta(N^{3/2})$ when k increases.

When legitimate parties are provided with quantum computers, we present two quantum protocols improving on the best known scheme before this work. Furthermore, for any $k \geqslant 2$, we give a quantum protocol in which legitimate parties establish a secret in $O(N)$ queries while the optimal quantum eavesdropping strategy requires $\Theta\big(N^{1+\frac{k}{k+1}}\big)$ queries, approaching $\Theta(N^2)$ when k increases.

Keywords: Cryptography, Quantum Algorithms, Quantum Lower Bounds, Random Oracle.

CONTENTS

To any *honest* perseverant ambitious person whose path to his/her objective is difficult, long, and possibly dark, but with some light here and there . . .

ACKNOWLEDGMENTS

First of all I give thanks for all the graces with which God (Allah) has blessed us, particularly for Its first creature, the *Mind*, and for making *different* nations.

This thesis is the fruit of more than four years of research under the supervision of Gilles Brassard and Louis Salvail. I am indebted to them for guidance, support and productive discussions. Particularly, I am thankful to Gilles for his support since he has been co-supervising my M.Sc. He is always willing to discuss and explain scientific and non-scientific matters (CV, LaTeX, etc.).

I am very grateful to my wonderful coauthors (Peter Høyer, Marc Kaplan and Sophie Laplante) for collaboration, which taught me a lot about quantum query complexity, and developed my professional skills

I deeply thank my examiner (Miklos Santha) for his thorough analysis of this thesis. His attention to details had remarkable impact on my work and some of my thinkings towards research. I also thank the other members of my thesis committee: Alain Tapp and Gena Hahn (and Pierre McKenzie).

I thank a lot the (including former) students at the laboratoire d'informatique théorique et quantique (LITQ) at Université de Montréal, who provided me a friendly research environment and enhanced my life experience and general culture: Alexandre Bibeau-Delisle, Michaël Cadilhac, Frédéric Dupuis, Hichem Eleuch, Yara Elias, Heinz Fiedler, Sébastien Gambs, Charles Hélou, Philippe Lamontagne, Rébecca Lapointe, Samuel Rannellucci, Benno Salwey, Dave Touchette (and possibly others). My office co-inhabitants (Claude Gravel, Abdulrahman Al-lahham) deserve special thanks. They provided me uncommon social experience and rich scientific environment.

Special thanks go to Mariette Paradis and the helpful employees, specially in the library.

Notable thanks go to friends who make my life less difficult and more beautiful.

Beyond doubt, this work would not have been possible without the support and trust of all my family.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Secrecy is a characteristic of mankind. People have always been concerned about protecting valuable or private information from unauthorized access. One common example is that of military commanders communicating with the troops or their superiors. Other examples include financial and medical information.

On top of that, the exchange of a massive amount of confidential information, such as electronic commerce and financial transactions, is taking place over the Internet. As a consequence, secrecy has become in peril and tools to maintain it become necessary. Techniques to remedy or at least to solve satisfactorily this problem belong to *cryptography*, which is nowadays the science of all aspects of privacy and information security, particularly secret communication and secret key establishment.

Following closely Stinson's classification [74], *key establishment* consists of any approach or technique that allows legitimate parties to establish a common secret key used to provide secure communication. Accordingly, we mainly distinguish three approaches: *key pre-distribution*, *session-key distribution* and *key agreement*. We describe these approaches later in a setting where there is a network of n users and possibly a trusted authority who is responsible for verifying the identities of users, transmitting keys, issuing certificates, etc.

In the next paragraph, we consider a cryptographic scenario that will be of multipurpose pedagogical benefits. More precisely, it illustrates how to establish secure communications over insecure channels despite efforts of any eavesdropper (or adversary), and shows the root of the problem of establishing secret keys. It also introduces the key agreement scenario that will be encountered from beginning to end of this thesis.

A typical cryptographic scenario is an everlasting story about three traditional parties to which we refer as machines or persons. Alice and Bob, also called the *legitimate parties*, want to communicate with each other over a public channel. However, they suspect that a third party **Eave** (for gender-neutral *eavesdropper*) is able to intercept their messages and for some reason determined to be privy to their communications. Therefore, they have decided to use the following standard cryptographic approach to preserve the confidentiality of their conversion.

When Alice wants to communicate with Bob, she does not send a message in plaintext that is readable by anyone. Instead, she transforms it by some means into a totally unreadable message called *ciphertext* (or cryptogram) in such a way that Bob can recover it but Eave cannot. Bob proceeds similarly to reply. The process of transforming plaintext into ciphertext is called *encryption* while the process of transforming ciphertext back into plaintext is called *decryption.* Both encryption and decryption are defined by a *cipher* (algorithm) and controlled by a cryptographic ingredient called the *secret key*, which Alice and Bob establish *somehow.* Devising such a "somehow" technique has been a major problem in cryptography: *How can Alice and Bob acquire this necessary random secret key?*

For centuries, people have solved this problem using the *key predistribution* paradigm requiring that (1) the channel is *authenticated*, meaning that legitimate parties are sure they are talking to each other; and (2) legitimate parties must agree on keys using secure channels prior to any secure communication. A *secure channel* might be a clandestine place or a trusted courier for instance. This approach has mainly been used in military and diplomacy applications in which communicants could meet occasionally in secret places to agree upon keys compiled into special books known as codebooks. Considering the Internet and wide use of electronic commerce, the second requirement has become a major limitation for the large-scale deployment of cryptography, and originated what is known historically as the *key "distribution" problem.* Nowadays, the *key establishment problem* is more accurate. We give a closer view on this problem by an example: if n entities want to communicate securely with each other, then there are $\binom{n}{2} = \frac{n(n-1)}{2} \approx \frac{n^2}{2}$ secret keys that must be generated, distributed, stored, and destroyed in a highly secure way. Even for moderately large n, the management of so many keys is infeasible in practice. This paradigm also suffers from another problem when keys are used in dynamic environments where new users join and others leave, which is usually impractical or simply too expensive to transmit keys over secure channels [67].

In a *session-key distribution* paradigm, an online trusted third authority (or key distribution centre) chooses session keys and distributes them to users who requested them via an interactive protocol. *Session keys* are secret keys used to encrypt information for a short period of time. These keys are sent encrypted by the trusted authority using the key predistribution paradigm, that is, the previously distributed secret keys to users. However, each entity must unconditionally trust the authority and it would be possible to compromise any communication via the centre. This approach is still typically used inside companies where each employee can get his "master secret key", which is used to derive session keys, in person when hired.

Finally, the *key agreement* approach refers to an interactive protocol between two parties that have no secret information in common and need to establish a secret key over a public authenticated channel. The major difference with key predistribution and session-key distribution is that the trusted authority is not needed any more.

The first unclassified document ever written to solve the secret key agreement problem and propose the notion of public key cryptography was a project proposal written by Merkle in 1974 [63]. Merkle demonstrated that the key pre-agreement requirement is *not* necessary by exhibiting a simple solution in such a fashion that when legitimate parties are willing to spend an amount of computation proportional to some parameter N, any classical eavesdropper needs to spend an amount of computation proportional to N^2 in order to obtain their established secret from the classical communicated messages. He only assumed the existence of a "one-way encryption function" [63], which can be formalized in the random oracle model that we explain after the following paragraph.

Shortly thereafter, Diffie and Hellman discovered a key agreement method that makes the cryptanalytic complexity "apparently exponentially" harder than the legitimate complexity [38]. However, even today, there is no formal security proof for their method. All relies on the *assumed* difficulty of a mathematical problem, namely on the conjectured difficulty of the discrete logarithm problem, which can be solved efficiently using a quantum computer thanks to Shor's algorithm [72]. In fact, no public-key scheme in the literature has any formal security proof, including RSA. In contrast, Merkle's scheme offers provable quadratic security against any possible classical attack under the assumption that the encryption function under consideration can be inverted only by exhaustive search.

Now, we identify our security framework. First of all, we consider the *random oracle model*, in which all parties have access to a *public* function f chosen uniformly at random from the set of all possible functions from a domain D into a range R. On any (new) input $i \in D$, this *black-box* (or oracle) function is evaluated in unit time by asking a question to a magic *black-box* that outputs (randomly) $f(i) \in R$. This interaction is called a *query*. It is very practical to think of such a function as a string (or table) of symbols y where each symbol was chosen independently and uniformly at random. If the domain size is S, then for all $1 \leqslant i \leqslant S$, this box outputs y_i on input i where $y_i \in R$ is a random value. Note that we can use binary oracles provided we disregard logarithmic factors. For the key agreement scenario, we assume that (1) the legitimate parties communicate through an *authenticated classical channel* on which eavesdropping is unrestricted; (2) all parties have access to the *same* oracle function and (3) a protocol is said to be *secure* whenever there is a *super-linear* computational gap between the complexities of legitimate parties and adversaries. In other words, for a fixed parameter N, legitimate parties make $O(N)$ queries while no adversary making $O(N)$ queries can learn the secret, except with $o(1)$ probability over the views of the protocol. A view determines a random run of the protocol in which Alice and Bob establish an n-bit secret using a random oracle. We vary over runs of any involved party: randomness of Alice, randomness of Bob, randomness of Eave, all possible choices of the n-bit secret and oracles.

It was a major open question in classical cryptography whether Merkle's method is optimal in the black-box model. In other words, is it possible to have a larger gap between the legitimate and eavesdropping complexities? In 1989, Impagliazzo and Rudich showed that any adversary making $O(N^6 \log N)$ queries can learn the key of every key agreement protocol in the random oracle model in which Alice and Bob make $O(N)$ queries [49]. In 2008, Barak and Mahmoody-Ghidary proved that a *quadratic* gap between the legitimate and eavesdropping efforts is the best possible in a classical world [8]. Sotakova [73] independently found a weaker result, showing that protocols with only one round of interaction (each party sends one message after querying the oracle) can achieve at most $O(N^2)$ security.

However, the situation is different in a world ruled by the quantum theory. Using *Grover's algorithm* [46], there is an obvious quantum attack that makes Merkle's scheme useless from a security point of view; the cryptanalytic task is as easy (up to constant factors) as the key agreement process.

Grover's algorithm (or its BBHT generalization [20]) is an extremely useful quantum algorithm since it can solve the following unstructured common search problem. For any integer K, we denote by $[K]$ the set of integers from 1 to K. Consider a black-box function $f : [N] \rightarrow \{0, 1\}$ with the promise that there exists one and only one $w \in [N]$ such that $f(w) = 1$. The (OR) problem is to find this w, which is then called a *solution*. Grover's algorithm solves this problem with bounded-error probability after $O(\sqrt{N}\,)$ quantum queries. Now, consider the same problem, however, with t solutions. In this case, it is crucial to use a generalization of Grover's algorithm, which can find a solution in $O(\sqrt{N/t}\,)$ quantum queries. Be aware that running (generalized) Grover's algorithm more than necessary may reduce the success probability down to zero!

It was commonly conjectured that secure key agreement in the random oracle model is *impossible*. Therefore, it is natural to examine the following question: *Can Merkle's scheme be made secure again if legitimate parties make use of quantum computations?*

A first solution was proposed by Brassard and Salvail in 2008 [23]. Their idea is essentially to consider Merkle's scheme, allow Bob to use a quantum computer and increase the function's domain size from N^2 to N^3 in such a way that any quantum eavesdropper needs $\Omega(\sqrt{N^3}) = \Omega(N^{3/2})$ queries [19]. Besides, it was conjectured that their scheme is optimal in this setting. A number of questions come to light:

1. Is there a protocol that requires more than $\Omega(N^{3/2})$ queries of any quantum eavesdropper if legitimate parties make use of quantum computations as well?
2. When legitimate parties only use classical computers, can any key agreement protocol in the random oracle model be broken with $O(N)$ quantum queries?
3. If the answer to Question 2 is *no*, then what is the optimal gap in this restrictive scenario?
4. Can the quadratic security of Merkle's scheme be restored if all parties make use of quantum powers?
5. When legitimate parties are empowered with quantum computers, can every key agreement protocol in the random oracle model be broken with $O(N^2)$ quantum queries?

The main challenge in this framework is that communications are taking place over a classical channel. Keep in mind that transmission of quantum information is forbidden. Another challenge is when legitimate parties are classical while the eavesdropper is allowed to use unrestricted quantum computations. Besides, the computation resource to which all parties have access is the same random oracle function. Therefore, throughout this study we will also be exploring the capabilities and limitations of all these resources, guided by the following questions.

6. How strong is the random oracle model in a quantum world?
7. Are quantum computers advantageous for cryptographers or eavesdroppers, considering key agreement in the oracle model?
8. Do quantum computers outperform their classical counterparts, considering the new problems raised in this work?

1.2 Contributions

We give in this thesis several novel key agreement protocols, answering some of the above open questions and making progress on others.

When Alice and Bob are empowered with quantum computers, we present two secure (quantum) protocols that improve on the scheme of Brassard and Salvail from $\Omega(N^{3/2})$ to $\Omega(N^{5/3})$ quantum queries, thus answering Question 1 *positively.* The first protocol is based on the element distinctness problem while the second is based on $\mathsf{2XOR}$. Without delay, we introduce these two problems.

Consider two positive integers $N \leqslant M$ and a black-box function $f : [N] \to [M]$. The *element distinctness* problem is to find a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $f(i) = f(j)$, or return $\nexists$ if they don't exist. Consider now a black-box function $g : [N] \to [M]$ and some $w \in [M]$. The $\mathsf{2XOR}$ problem is to find a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $g(i) \oplus g(j) = w$, or return $\nexists$ otherwise. The operator $\oplus$ denotes the *bitwise exclusive-or.* Note that the decision versions of the search problems are usually considered in complexity theory when proving lower bounds.

When Alice and Bob are restricted to use *classical* computers, we present the *first* classical protocol secure against any quantum adversary. Indeed, this scheme forces any quantum eavesdropper to ask $\Omega(N^{13/12})$ quantum queries before being able to learn the secret key, which answers Question 2 *negatively.* Furthermore, we present an improved secure scheme providing security of $\Omega(N^{7/6})$ quantum queries. As was the case in our quantum protocols, the $\Omega(N^{13/12})$ scheme is based on element distinctness while the $\Omega(N^{7/6})$ scheme is based on $\mathsf{2XOR}$.

Furthermore, we made progress on Questions 3 and 4. We have discovered two sequences of protocols denoted by Q_k and C_k, for any fixed integer $k \geqslant 2$, with the following properties.

In protocol Q_k, a classical Alice establishes a secret with a quantum Bob after $O(N)$ accesses to a random oracle in such a way that the optimal quantum eavesdropping strategy requires of the eavesdropper to query the same random oracle $\Theta\big(N^{1+\frac{k}{k+1}}\big)$ times, thus approaching $\Theta(N^2)$ in the limit. Therefore, key agreement protocols in the random oracle model can be arbitrarily as secure in our quantum world as they were known to Merkle in 1974. Thus, almost quadratic security can be restored in a quantum world.

In protocol C_k, classical Alice and Bob establish a secret after $O(N)$ queries to a random oracle in such a way that the optimal quantum eavesdropping strategy requires $\Theta\big(N^{\frac{1}{2}+\frac{k}{k+1}}\big)$ queries, which tends to $\Theta(N^{3/2})$ when k increases. Hence, classical Alice and Bob can agree on a secret key against any quantum eavesdropper with as good a security (in the limit) as it was known to be possible before when Alice and Bob are enabled with quantum computers [23].

1.3 Related Work

We mention in this section selected work, directly useful or related somehow to our research topic, grouped into several categories. We also define several problems of particular importance not only in this work but in classical and quantum computer science in general. The search versions of these problems characterize better our cryptographic scenario. However, their decision versions are often considered in complexity theory since a lower bound for a decision problem is simpler to prove and implies directly a lower bound for its search version.

Collision and Element Distinctness Problems. A *collision* for a function f is a pair of distinct elements i, j such that $f(i) = f(j)$. Deciding whether a collision occurs in a function f is equivalent to deciding whether f is injective, that is, whether f is one-to-one, which is the element distinctness problem. The problem is important in cryptography, because it enables modelling fundamental primitives (*hash functions*), and of general importance in classical and quantum computer science. A function $f : [N] \to [M]$ is said to be *r-to-one*, for an integer r dividing N, if every element in its image has exactly r pre-images.

The first quantum algorithm that finds collisions in r-to-one functions was given by Brassard, Høyer and Tapp [25], providing novel applications of Grover's algorithm. When f is r-to-one, their algorithm finds a collision after $O(\sqrt[3]{N/r}\,)$

expected function evaluations. In particular, when f is *two-to-one*, this algorithm finds a collision after $O(\sqrt[3]{N}\,)$ evaluations. In contrast, $\Theta(\sqrt{N}\,)$ evaluations are necessary and sufficient for a classical algorithm to find a collision even allowing randomization. They also presented a quantum algorithm for claw-finding. A *claw* in functions f and g having the same range is a pair of elements i, j such that $f(i) = g(j)$, which is closely related to the notion of collisions.

Inspired by these algorithms, Buhrman, Dürr, Heiligman, Høyer, Magniez, Sántha and de Wolf [32] gave several applications of *amplitude amplification* [26] to find collisions and claws in unrestricted functions, yielding an $O(N^{3/4} \log N)$ collision-finding quantum algorithm, which implies an $O(N^{3/4} \log N)$ upper bound for element distinctness. They also gave an $\Omega(\sqrt{N}\,)$ quantum lower bound for element distinctness, using a reduction from the OR problem. The complexity measure they used is the *number of comparisons* between elements. However, all their bounds remain the same up to logarithmic factors if they want to count the number of function evaluations instead. In the *comparison-based oracle model*, when we query a given black-box function f on (i, j), the oracle outputs the truth-value (0 or 1) of the statement "$f(i) \leqslant f(j)$".

Using the *polynomial method* [10], Aaronson and Shi [2] proved that any quantum algorithm for finding collisions in r-to-one functions must evaluate the function $\Omega(\sqrt[3]{N/r}\,)$ times, matching the upper bound of Brassard, Høyer and Tapp [25]. Thus, this is optimal – even for the decision version of this problem. Furthermore, they derived a lower bound of $\Omega(N^{2/3})$ queries for element distinctness, using a reduction from a variant of the collision problem. Their paper is in fact the fruit of two separate closely related earlier publications [1, 71].

In the other direction, Ambainis gave an $O(N^{2/3})$ quantum query algorithm [4] for element distinctness, which proceeds by quantum walks on Johnson graphs, improving on the $O(N^{3/4})$ algorithm [32] and matching the $\Omega(N^{2/3})$ lower bound [2].

The proof of Aaronson and Shi [2] is applicable only when the range size M of the function is such that $M \geqslant 3N/2$. If $M = N$, the lower bound becomes $\Omega(\sqrt[4]{N})$. Fortunately, Ambainis [6] and Kutin [53] independently removed this constraint.

In fact, Ambainis gave a general method to prove lower bounds for problems with small range. He proved that for any symmetric property (or problem) defined on some function $f : [N] \to [M]$ the polynomial degree is the same for any $M \geqslant N$. In other words, for any function of range size M, the quantum query lower bound proved using the polynomial method implies immediately the same lower bound for any $M \geqslant N$. A property f is *symmetric* if it remains the same if we permute the inputs before applying f or permute the outputs after applying f. Therefore, the lower bounds for collision and element distinctness remain valid even with small range, since both problems are symmetric. Hence, element distinctness is solved optimally in $\Theta(N^{2/3})$ quantum queries without restrictions on the range. Eventually, Belovs proved the lower bound for element distinctness [13] using the *negative* (or *generalized*) *adversary method* [80, 81]. Notice that classical lower bounds for element distinctness have been known since decades [45, 57, 65].

In the aforementioned paper [4], Ambainis also gave an $O(N^{\frac{k}{k+1}})$ quantum algorithm for element k-distinctness problem, which is to decide whether or not there exist k pre-images mapped to the same image under a given function f. It is a generalization of his element distinctness algorithm and related to the optimal quantum attacks against our generalized protocols described in Chapter 6. He left open whether his algorithm for k-element distinctness is optimal. Answering this question negatively, Belovs provided a more efficient algorithm [14] having query complexity $O(N^{1-2^{k-2}/2^k-1})$ or more compactly $o(N^{3/4})$.

Shortly after Ambainis's paper, Childs and Eisenberg observed that Ambainis's quantum algorithm for k-element distinctness is much more general than what it was designed to resolve. More precisely, this algorithm gives the means to solve *any* problem that can be modelled as the subset-finding problem having any given property. Indeed, they mentioned several related applications, for instance finding a set of k consecutive function values, relatively prime function values, k-clique in an N-vertex graph, and variants of the kSUM problem. They also provided a much simpler query-complexity analysis than that of the original paper [4]. On the other hand, they left open which of their algorithms for subset finding is or are optimal.

kSUM. In addition to element distinctness and 2XOR, the kSUM problem is essential in this thesis. Let s be an arbitrary element of a finite abelian group $\mathbb{G}$ and consider a positive integer k. Given elements $x_1, \ldots, x_N$ of $\mathbb{G}$, the kSUM problem is to find a subset of k elements that sum to s, or return $\nexists$ if such subset does not exist. Actually, any problem based on the $\oplus$ is special case of kSUM, which is in turn related to element k-distinctness.

Belovs and Špalek [15] proved that the subset-finding quantum algorithm for kSUM [36] is optimal, using the generalized adversary method (or bound). Except for this work, we don't know of any other cryptographic application considering the kSUM problem in the random oracle model. However, 3SUM has been well studied and a long-standing problem in the classical time-complexity literature. The best known algorithm takes $\Theta(N^2)$ time [42], commonly believed to be optimal.

Oblivious Transfer à la Merkle. The notion of oblivious transfer was originally introduced in Wiesner's paper [77], which served as inspiration for the invention of quantum key distribution [16]. Unfortunately, it was only published in 1983, making a more pathetic story than that of Merkle's scheme [63]! However, in 1981 Rabin was the first to publish it [68]. The original version of oblivious transfer (OT) is a formalization of an erasure channel with probability 1/2 and can be described as follows. Alice (the *sender*) sends a bit b to Bob (the *receiver*) via an OT machine. With probability 1/2, Bob receives b and with complementary probability he receives the symbol $\perp$ (nothing), which is just an evidence that a bit was sent but the information was lost during the communication. Note that Alice does not know whether or not her bit was received. An equivalent primitive is *1-out-of-2* oblivious transfer, which was invented in 1985 by Even, Goldreich, and Lempel [41]. In this variant, Alice has two secret messages (instead of one bit) and Bob receives one of them at random without gaining any information about the other message. Again Alice should not know which message Bob received. More formally, Alice inputs two secret messages m_0 and m_1 into the OT machine that flips a coin and sends one of the messages with equal probabilities.

Be aware that the *1-out-of-2* oblivious transfer has been used in the literature slightly differently: Alice inputs two secret messages m_0 and m_1 into the OT machine to which Bob inputs one bit b to indicate which input he would like to receive. The machine outputs m_b and discards m_{1-b}. Summing up, at the end of the protocol, Alice cannot learn any information about b and Bob learns m_b but nothing about the other message.

Shown to be a universal primitive for two-party computation by Kilian [52], oblivious transfer is fundamental in cryptography. A box that implements a perfect oblivious transfer between two participants would give them means to achieve an arbitrary two-input computations with information-theoretic security.

Merkle's idea was used beyond key agreement by Brassard, Salvail and Tapp [28]. Considering the birthday problem ("paradox") and one-way permutations, they introduced a classical oblivious transfer scheme and proved that a classical cheater requires $\Theta(t^{3/2})$ permutation queries, where t is the legitimate amount of queries needed to implement it. A quantum adversary, however, breaks this scheme after $O(t^{5/6})$ queries, which is more efficient than running the legitimate protocol! Allowing honest parties to use quantum computers, they also present another scheme against which their best quantum attack makes $O(t^{4/3})$ queries. It is still an open question whether this attack is optimal.

Bounded-Memory Model. The *bounded-storage* (or *bounded-memory)* model was proposed by Maurer [61] to achieve provable cryptographic schemes even against adversaries with unlimited computational resources. There is no computational hardness assumption like the prime factoring or discrete logarithm problems. The only assumption is that the adversary's memory size is bounded by a value s.

The main idea is the following. A random t-bit string R is available *temporarily* to all parties, and can be broadcast by a satellite, transmitted over a network or stored on a public high-density storage media. The string R is much larger than the adversary's memory capacity, meaning that he can store only partial information about R. However, he can use unlimited computing power to calculate any probabilistic function $f : \{0,1\}^t \to \{0,1\}^s$. As long as the function's output size does not exceed the available memory, security in the above sense is maintained.

More precisely, Alice and Bob randomly select a small subset of R and each stores these values, afterwards R disappears. In this model, legitimate parties initially share a *random short secret key* that determines which bits of R they need to access and how they must process these bits in order to derive a longer secret key. Because of the random selection of the subset and Eave's memory size, he may have at most partial knowledge about this fraction of R. Therefore, legitimate parties can apply *privacy amplification* [18] in order to distill a perfectly secret key, that is, generate a key about which Eave has essentially no information even using unlimited computing power. Note that even after privacy amplification, which reduces the input key, the final secret key is much longer than the initial key. Summing up, the goal in this model is *key expansion* rather than establishing secret keys.

The *bare bounded-storage model* is another framework that was proposed by Cachin and Maurer [33]. In this model, legitimate parties who share *no initial key* proceed as follows. Each stores a random subset of $O(\sqrt{t}\,)$ bits of a random t-bit string R, which is much more than the selected subset in the previous model. After R disappears, they agree on the commonly chosen bits, which exist considering the birthday problem. On the other hand, with overwhelming probability, Eave has only partial information about these bits so that legitimate parties can apply *privacy amplification* [18] to distill a perfectly secret key.

Dziembowski and Maurer [40] proved that secure key agreement in the bare bounded-memory model is impossible unless legitimate parties have memory size in $O(\sqrt{s}\,)$, which is so large that the practicality of this approach (without an initially shared short key) is inherently limited. This is the same optimal quadratic gap between the number of queries of legitimate parties and that of eavesdroppers in the random oracle model. Cryptography in the bounded quantum-memory model was considered by Damgård, Fehr, Salvail and Schaffner [37].

Quantum Computation. Our proposed protocols and the related problems may entail understanding the capabilities and limitations of quantum computers. Indeed, it is widely believed that quantum computing is promising. However, building a quantum computer faces theoretical as well as technological challenging obstacles, in particular providing convincing arguments that a quantum computer is much more useful than a classical counterpart. These arguments mainly are (1) designing quantum algorithms significantly more efficient than their classical counterparts; (2) identifying old and/or new problems that give the means to understand the power and limitations of the quantum computing theory such as *pseudo-telepathy* [27, 30]; and (3) discover ideas that have no classical counterpart such as *entanglement*, *teleportation* and quantum key distribution. Our schemes are useful in the sense of the first two motivations because they provide problems for which quantum algorithms are provably better than their classical equivalents.

Shor's and Grover's algorithms are the most important quantum algorithms at the time of writing. The first one has at least two major known applications: solving the prime factoring and discrete logarithm problems in polynomial time. However, Grover's algorithm and its generalization have many applications since plenty of important problems in computer science can be reduced to search problems. Such problems range from sorting to graph colouring to attacking cryptographic protocols. Merkle's schemes, their quantum variant and our new protocols can also be reduced to search problems.

1.4 Outline

The remaining material of this thesis is organized as follows. Preliminaries are given in Chapter 2, and Merkle's scheme as well as its quantum variant are discussed in Chapter 3. Our contributions are divided into Chapters 4, 5 and 6. The first consists of two protocols in the quantum setting while the second contains two protocols in which the legitimate parties are restricted to classical computation. Chapter 6 contains our two families of improved generalized protocols. Finally, we conclude and raise some open questions in Chapter 7.

CHAPTER 2

PRELIMINARIES

2.1 Notations and Notions

For any positive integer K, the set $[K]$ consists of all integers from 1 to K and $[K]'$ denotes $[K] \cup \{0\}$. The set $\{0,1\}^n$ denotes the set of all possible n-bit stings. Generally, given a non-empty set S, we denote by S^n the set of all possible strings of exactly n elements of S. We denote by $\{0,1\}^*$ the set of all *finite* binary strings and $\{0,1\}^\infty$ the set of all *infinite* ones. The set of natural numbers is denoted by $\mathbb{N}$ while $\mathbb{R}^+$ denotes the set of non-negative real numbers. The notation $|\cdot|$ is used to denote the length of a string or the size of a set, depending on the context.

A *probabilistic algorithm* has access to uniform random bits (unbiased coin flips). Equivalently, a randomized or probabilistic algorithm is given, in addition to its input, a uniformly distributed bit-string of sufficient length (see Section 2.10).

An algorithm $\mathcal{A}$ is said to run in *polynomial time* if there is a polynomial $p(\cdot)$ such that for every input x, its output $\mathcal{A}(x)$ terminates within at most $p(|x|)$ steps.

Definition 2.1.1 (Negligible Function). *A function $f : \mathbb{N} \to \mathbb{R}^+$ is* negligible *in n if for every real constant $c > 0$ there exists n_c such that $f(n) < \frac{1}{n^c}$ for all $n \geqslant n_c$.*

In other words, *negligible* describes any function that decreases faster than the inverse of any positive polynomial. This definition is meaningful in standard cryptography (without oracle), in which the required level of security is at least super-polynomial. However, the security level in our context (oracle model) is polynomial. Therefore, what we need is the notion of vanishing probability instead.

Definition 2.1.2 (Vanishing Function). *A function $f : \mathbb{N} \to \mathbb{R}^+$ is* vanishing *in n if for every real constant $c > 0$ there exists n_c such that $f(n) < c$ for all $n \geqslant n_c$.*

Equivalently, a vanishing function is in $o(1)$. We might also specify how fast a function tends to zero. For instance, the function $1/N^2$ is quadratically vanishing.

2.2 Random Function

We consider throughout this thesis the set (or family) of all functions from D to R, denoted by $\mathsf{Func}(D, R)$ or $\mathsf{Func}(\ell, L)$ for convenience. Specifically, we are interested only in functions whose domain D and range R are finite. Formally, we consider $D = \{0,1\}^\ell$ and $R = \{0,1\}^L$ for some integers $\ell, L \geqslant 1$ representing the *input length* and the *output length*, respectively.

In this framework, the two notions "random function" and "choose a function at random" are equivalent [11], which we explain immediately and differently. Assume that each element $f \in \mathsf{Func}(D, R)$ is uniquely identified by an index $i \in I$, where I is the finite set of all possible indices. There is a (uniform) probability distribution on the set of indices, thus inducing the same probability distribution on $\mathsf{Func}(D, R)$. To choose a function at random, we choose an index $i \in I$ at random and then consider the element (instance) f_i. The equivalence of the two notions is so crucial to understand that we encapsulate them in the following definition.

Definition 2.2.1 (Random functions)**.** *Let $D = \{0,1\}^\ell$ and $R = \{0,1\}^L$ be two non-empty sets, where $\ell, L \geqslant 1$ are two positive integers. Consider the set of all functions from D to R, denoted by $\mathsf{Func}(D, R)$ or $\mathsf{Func}(\ell, L)$. A* random function *mapping ℓ-bits to L-bits is an element of $\mathsf{Func}(\ell, L)$ chosen uniformly at random.*

Classically, a good way to think about a random function may be the following: *the value assigned to any new point in the domain is randomly and independently selected from the range.* However, this might be *misleading* or a *bad* viewpoint in the quantum framework as we will explain later.

It will be useful to compute the size of the set $\mathsf{Func}(\ell, L)$. To specify a function it is necessary and sufficient to provide its value on any element in its domain. More precisely, think about a function $f \in \mathsf{Func}(\ell, L)$ as a huge look-up table such that each row contains one possible input x coupled with its image $f(x)$. Since there are 2^ℓ possible inputs and each input needs L bits to specify the output value, any function can be represented by $2^\ell \cdot L$ bits. Hence, there are $2^{2^\ell \cdot L}$ possible functions in total.

2.3 Random Oracle

Oracles are a very useful tool to provide an idealized implementation of a function. Although defined and used differently in complexity theory and cryptography, they share the following properties in both fields:

1. *Abstraction*: think about an oracle as a *magic box* that answers well defined questions.
2. *Efficiency*: the oracle's answer to any point in the domain is computed in unit time (or perhaps in polynomial time in computational complexity).

Informally, a *random oracle* is an oracle incorporating randomness in its outputs. There are two main formal definitions for random oracles, one in computational complexity by Bennett and Gill [17] and other in complexity-based cryptography.

Definition 2.3.1 (Random oracles in complexity theory). *A random oracle associates the result of a coin toss to each string, that is a map* $\mathcal{O} : \{0,1\}^* \to \{0,1\}$.

Oracles in cryptography were first used by Brassard [21], as pointed out by Impagliazzo and Rudich [49]. However, the random oracle paradigm was formalized and defended by Bellare and Rogaway [12], who defined a random oracle as a map $\mathcal{O} : \{0,1\}^* \to \{0,1\}^\infty$ to avoid fixing its output length. Inspired by [12, 49], we give the following definition, capturing the *integer* random oracles.

Definition 2.3.2 (Random oracles in cryptography). *A random oracle is a map* $\mathcal{O} : \{0,1\}^\ell \to \{0,1\}^{L(\ell)}$ *that assigns, to each* (new) *input, an integer* (chosen uniformly at random) *from a finite range whose size is a function* L *of an* $\ell \geqslant 1$.

2.4 Black-Box Function (Random Function Oracle)

With each random oracle we associate a function from ℓ-bit strings to L-bits strings. As we vary over all possible oracles $\mathcal{O}$, we get all corresponding functions, "each with the same frequency" [49], thus we get the family of functions $\mathsf{Func}(\ell, L)$. The notions of random oracles and random function oracles will be used interchangeably.

We treat any random instance $f \in \mathsf{Func}(D, R)$ intuitively in the following sense. The function f is placed in a box impossible to look inside, and evaluated on any input *only* by asking questions to this box, which returns $f(x)$ on an input x. This access or interaction is referred to as *querying the box* (or f) on x, which is itself called the *query* x to the function. In cryptography, any query is assumed to be computed in a unit time.

It might be useful to think of the function f as black-box that contains a tuple $y = (y_1 \dots y_{2^\ell})$ and outputs y_i on input i where y_i is chosen randomly in R.

Needless to say, although we consider integer oracles, our analyses also remain valid for binary oracles, provided we disregard logarithmic factors. An oracle may be *classical* or *quantum*. Queries in the quantum setting must be reversible and are often done in superposition.

Choosing a function f at random implies that the box is programmed to output answers according to f. More precisely, we assume that the *full specification* of this function is available in the box once it is chosen. This view is always considered in the quantum computation context, where queries are often done in superposition. Though, there is another classical dynamic view: queries are evaluated one by one.

2.5 Random Oracle Model

The random oracle model in cryptography was introduced to be a bridge between theoretical cryptography and practical one for the sake of designing secure schemes. This methodology mainly consists of two steps.

First, one designs and proves the security of an ideal system in which all parties (including adversaries) have access to an oracle, that is, a public uniformly-chosen function in a particular family of functions. Second, once the first step is fulfilled, one replaces the random oracle by a public "secure cryptographic hash function". For more details refer to the paper of Bellare and Rogaway [12], who explicitly formulated and defended this popular methodology. Another reference that covers well this topic is the class notes of Bellare and Rogaway [11].

2.6 Key Agreement

The task we will study in this thesis is the secret-key agreement in the random oracle model, which we define here for our convenience. Notice that key agreement is one of the key establishment approaches reviewed in the introduction.

A *key agreement protocol* is an interactive protocol between a pair of two probabilistic linear query algorithms (or machines) Alice and Bob, having no-secret information in common, to establish a secret over a public authenticated channel. Each party has a set of private tapes: a *random-bit* tape, an *input* tape, a *work* tape and a *secret* tape. In addition, they have a common communication channel, which both can read and write. We consider only protocols restricted to *one round* of interaction, meaning that each legitimate party queries the oracle a number of times proportional to some fixed parameter, makes some computations, and sends a message to the other party.

A typical *run* of the protocol can be viewed as follows: Alice and Bob receive a security parameter n and are given access to a black-box function. Next, Alice queries the oracle on random distinct points, communicates via the channel and writes an n-bit string on her secret tape. Bob does similarly. If the secret strings are the same, then Alice and Bob are said to *agree.* The entire history of their writings on the channel is known as the *conversation* or the *transcript.*

On the opposite side, a probabilistic linear query algorithm Eave (eavesdropper) is determined to learn the secret, having full knowledge of the conversation (between Alice and Bob) and access to the same black-box function.

For convenience, we refer to Alice and Bob as persons occasionally. However, Eave, which replaces the traditional Eve, is gender-neutral (or a daemon).

Note that the assumption "authenticated channel" is unavoidable in any key agreement scenario. It ensures that any adversarial attempt to modify a message or inject a new one into the conversion can be detected by legitimate parties. Establishing secret keys over a public *unauthenticated* channel is impossible! This section and the next one is closely based on Impagliazzo and Rudich's work [49].

2.6.1 View of Protocol

We investigate a random world where Alice and Bob try to establish an n-bit secret. More precisely, we vary over runs of Alice, Bob, Eave, and oracles. Formally, a "world situation" or *view* is 5-tuple $(n, r_a, r_b, r_e, \mathcal{O})$. The first entry n is the input, and $\mathcal{O}$ is a random oracle. The secret random-bit tapes r_a, r_b, r_e are assigned to Alice, Bob and Eave, respectively. Let $\mathcal{V}_n$ be the set of all views where Alice and Bob attempt to agree on an n-bit secret. One might think of $\mathcal{V}_n$ as a probability space with the uniform distribution. A view determines a random run of the protocol using a random oracle.

2.7 Security Model

Security definition of a cryptographic scheme may consist of three parts: 1) define the notion "*secure*" or equivalently its negation "*break*"; 2) specify the computation power of all involved parties as well as the computation resources (e.g., oracle) to which they have access; and 3) identify the type or level of required security.

In this section we decide on the security framework of secret-key agreement protocols in the random oracle model, which is different from that of standard cryptography. To understand better the former, it is useful to introduce first the well-known framework of standard cryptography, picking out the key agreement task as our running example for this purpose.

2.7.1 Computational Complexity Model

Current standard cryptography is based on the fundamental assumption P $\neq$ NP in computational complexity [44], thus it is called complexity-based cryptography. More precisely, a legitimate cryptographic algorithm runs *efficiently* (polynomial time). Nevertheless, an adversary is not able to break this scheme unless given at least a *super-polynomial time.* A scheme is said to be "*broken*" if an adversary can know a part of the established secret key or even distinguish it from a randomly selected key.

Rooted in complexity theory, this framework inherits three essential notions: the *asymptotic* approach, *efficient* or *feasible* algorithms, and *negligible* functions. In complexity-based cryptography any scheme has a security parameter n so that the running time of the honest parties, the running time of the adversary, and the adversary's success probability are all *measured asymptotically in terms of* n. This important property is useful to make formal and rigorous security proofs. An *efficient algorithm* is a deterministic or probabilistic polynomial algorithm whose time complexity is polynomial in n. A *negligible function* in n is smaller than the inverse of any positive polynomial (Definition 2.1.1).

Having clarified all the important notions, we specify the framework of the contemporary applied cryptography:

1. Legitimate parties initialize the cryptographic scheme after fixing a value for its security parameter n whose value is assumed to be known by the adversary.
2. Legitimate parties run efficiently.
3. Security is maintained only against efficient adversaries. Clearly, any scheme can be broken by a super-polynomial time strategy, which is so unfeasible that such threats are not considered. That is why this approach provides only computational security, contrasting with information-theoretic security based on the fact that adversaries do not have enough information regardless of their computational resources power (see more in Stinson's textbook [74]).
4. An efficient adversary succeeds in breaking any scheme *only* with *negligible* probability.
5. A scheme must be secure in most cases or have "average-case hardness".
6. An adversary should not be able even to distinguish the established secret key from a truly random key except with *negligible* probability.

Modern cryptography is well covered in the textbooks [44, 60, 62, 74]. Now, we give a formal definition of the standard notion of security.

Definition 2.7.1 (Asymptotic security)**.** *A scheme is secure if no probabilistic polynomial-time adversary can break the scheme except with negligible probability.*

2.7.2 Random Oracle Model

Here is the security framework in the random oracle paradigm, after making two essential relaxations of the notions of standard cryptography, namely efficient and negligible.

1. Legitimate parties decide on the security parameter and a random function oracle (chosen as in Section 2.2) whose domain size is a function of some parameter N. Related to the oracle, the value of N is assumed to be public and the function is accessible to all parties.
2. Legitimate algorithms are *linear query* ones, meaning that each party asks $O(N)$ queries.
3. Security is maintained only against *linear query eavesdroppers*: any scheme can be broken by an adversary asking the oracle a super-linear in N queries. Comparing with standard cryptography, an "efficient" algorithm is one whose query complexity is linear rather than polynomial. It is crucial to keep this in mind. A protocol is *secure* if no probabilistic linear query algorithm Eave can break it (guess the secret) except with $o(1)$ probability (this is vanishing). This notion is meaningful since any classical protocol in the oracle model can be broken in $O(N^2)$ queries [8]. In the quantum case it is even worse.
4. A linear query adversary $\mathcal{A}$ can succeed in breaking any scheme *only* with $o(1)$ probability. Thus, repeating $\mathcal{A}$ a constant number of times does not harm the security. However, repeating it a *sub-linear* or even *logarithmic* number of times can amplify the success probability as close to 1 as desired.
5. We require a scheme to be hard to break in most cases or on the average ("average-case hardness"), similarly to standard cryptography.
6. In contrast with standard cryptography, the security concern here is that *no* adversary can find the secret or even a *part* of it.

Definition 2.7.2 (Security in the random oracle model)**.** *A scheme is secure if no probabilistic linear query algorithm is able to break it except with* $o(1)$ *probability over all the possible random oracles under consideration.*

2.8 Common Problems and Their Query Complexities

We review in this section several problems inherently related to our protocols. Although we consider the search versions in the cryptographic context, decision versions are examined to prove lower bounds. In the following material, N and M are positive integers.

2.8.1 The Collision Problem and Its Generalization

Consider a function $f : [N] \to [M]$ and an integer $r > 1$ with r dividing N. We say that f is *r-to-one* if every element in its image has exactly r preimages. A *collision* for f consists of two distinct elements $i, j \in [N]$ such that $f(i) = f(j)$.

Definition 2.8.1 (The r-to-one collision problem)**.** *Let f be a black-box function with the promise that it is either one-to-one or r-to-one. The* r-to-one collision problem *is to distinguish between these two cases. In particular, the* collision problem *is to decide whether f is two-to-one or one-to-one.*

Importantly, the collision problem reduces to element distinctness [1, 2]. Briefly, if the first problem requires T queries, then element distinctness requires T^2 queries.

Theorem 2.8.2 (Lower bound [2])**.** *Let $n > 0$ and $r \geqslant 2$ be integers with $r|n$, and let a function of domain size n be given as an oracle with the promise that it is either one-to-one or r-to-one. Then any error-bounded quantum algorithm for distinguishing these two cases must query the function $\Omega((n/r)^{1/3})$ times. Thus, finding a collision in an r-to-one function of domain size n requires $\Omega((n/r)^{1/3})$ queries.*

2.8.2 The Element Distinctness Problem and Its Generalization

Definition 2.8.3 (Element distinctness)**.** *Given a black-box function $\xi : [N] \to [M]$, the element distinctness problem* (ED) *is to decide whether there exists a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $\xi(i) = \xi(j)$.*

This problem can be generalized to what is known as the element k-distinctness or k-element distinctness.

Definition 2.8.4 (k-Element distinctness)**.** *Given as input a black-box function $\xi : [N] \to [M]$, the k-element distinctness problem is to decide whether or not there exists k indices $i_1, \dots, i_k$ for which $\xi(i_1) = \xi(i_2) = \cdots = \xi(i_k)$.*

Theorem 2.8.5 (Upper bound [4])**.** *Element k-distinctness can be solved by a quantum algorithm with $O(N^{k/(k+1)})$ queries. In particular, element distinctness can be solved in $O(N^{2/3})$ quantum queries.*

Belovs designed a more efficient algorithm for element k-distinctness as stated by the following theorem.

Theorem 2.8.6 (Upper bound [14])**.** *For an arbitrary fixed integer $k \geqslant 2$, the element k-distinctness problem can be solved by a bounded-error quantum algorithm in $O(N^{1-2^{k-2}/2^k-1})$ quantum queries.*

2.8.3 The Subset-Finding Problem

Childs and Eisenberg [36] proved that Ambainis's quantum algorithm for k-element distinctness [4] actually solves the *subset-finding* problem (SF), which is very useful.

Suppose that we are given as input a black-box function $f : D \to R$ and a property $\mathcal{P} \subset (D \times R)^k$ where D and R are finite and $|D| = N$. The problem is to output some k-subset $\{x_1, \dots, x_k\} \subset D$ such that $((x_1, f(x_1), \dots, (x_k, f(x_k))) \in \mathcal{P}$.

Theorem 2.8.7 (Upper bound [36])**.** *The quantum query complexity of k-subset finding is $O(N^{k/(k+1)})$.*

2.8.4 The kXOR Problem

Definition 2.8.8 (kXOR problem)**.** *We are given as input a black-box function $\xi : [N] \to [M]$ and some $w \in [M]$. The* kXOR *problem is to decide whether or not there exists k indices $i_1, \dots, i_k$ for which $\xi(i_1) \oplus \xi(i_2) \oplus \dots \xi(i_k) = w$.*

The case $k = 2$ is specially important since we use it in two of our protocols and reduce element distinctness to it in Section 4.2.3. Therefore, it is worth having a separate definition.

Definition 2.8.9 (2XOR problem)**.** *We are given as input a black-box function* $\xi : [N] \to [M]$ *and some* $w \in [M]$*. The* 2XOR *problem is to decide whether or not there exists a pair* (i, j)*,* $1 \leqslant i < j \leqslant N$*, for which* $\xi(i) \oplus \xi(j) = w$*.*

2.8.5 The kSUM Problem

We first give the definition to which the lower bound theorem [15] is applied. The general definition follows.

Definition 2.8.10 (kSUM)**.** *Consider an arbitrary element* t *of a finite abelian group* $\mathbb{G}$*. For any given positive integer* k*, the* kSUM *problem is to decide whether the elements* $x_1, \ldots, x_N \in \mathbb{G}$ *contains a subset of* k *distinct elements that sum to* t*.*

Theorem 2.8.11 (Lower bound for kSUM [15])**.** *For a fixed positive integer* k*, the quantum query complexity of the* kSUM *problem is* $\Omega(N^{k/k+1})$ *provided that* $|\mathbb{G}| \geqslant N^k$*.*

Here is another definition that we might encounter in the literature.

Definition 2.8.12 (kSUM's another definition)**.** *We are given as input a black-box function* $\xi : [N] \to [M]$ *and some* $w \in [M]$*. The* kSUM *problem is to find* k *indices,* $i_1, \ldots, i_k$*, for which* $\xi(i_1) + \xi(i_2) + \cdots + \xi(i_k) = w$*, or return* $\nexists$ *if they don't exist.*

2.8.6 The Birthday Problem (or "Paradox")

Consider any set X containing N elements such that $\Pr(x) = 1/N$ for all $x \in X$. Pick randomly and with replacement n elements of X. What is the probability that at least two of these n elements will be the same? This event is commonly referred to as *collision* [74] and denoted by Coll.

This problem is solved using a probability argument analogous to the "birthday paradox", which says that in a group of 23 randomly chosen persons, at least two will have the same birthday with probability greater than 1/2. Although it probably seems counter-intuitive, it is not a paradox at all after calculations. Therefore, we will refer to it as the *birthday problem*.

The upper bound on this probability will be invoked frequently. Therefore, we write it explicitly:

$$\Pr[\mathsf{Coll}] \leqslant \frac{n \cdot (n-1)}{2N} < \frac{n^2}{2N}.$$

This inequality can be obtained taking into account the $n \cdot (n-1)/2$ possible pairs (count only pairs of distinct elements) and using the union-bound.

We equally need to have an expression that relates all the involved parameters. The probability of finding at least one collision is estimated to be at least

$$1 - e^{\frac{-1}{2}(n-1)n/N} \approx 1 - e^{\frac{-n^2}{2N}}.$$

Denoting this probability by ε, we can find a good approximation that relates ε, n and N, which is formulated as follows:

$$n \approx \sqrt{2N \ln \frac{1}{1-\varepsilon}}.$$

Therefore, for

$$n > \sqrt{2 \ln 2}\,\sqrt{N} \approx 1.17\sqrt{N},$$

at least two outcomes will be the same with probability at least 1/2. This problem is well proved and presented from a cryptographic viewpoint by Stinson [74].

Being frequently interested in avoiding collisions in functions, we need to set a lower bound on the range size. We achieve this task in detail in Section 5.3.4. Finally, observe that there are several manifestations of this important problem. For instance, the study of collisions between two subsets is discussed by Joux [50].

2.9 Computing Partial Boolean Function

We consider *partial functions* or *promise problems*, which are decision problems for which the input is promised to be drawn from a subset of the function domain. Partial functions are useful and general enough to achieve our computation goals. Usually lower bounds for partial functions are easier to find and imply directly lower bounds for general functions.

Let $f \in \mathsf{Func}(D, R)$ be an oracle function, and let $D = [N]$ and $R = [M]$ for convenience. The objective is to determine *whether or not* f has a *specific property.* For example, "Is f one-to-one or two-to-one?" or "Are there k pre-images under f that sum to some given integer w?". More formally, we would like to compute a partial Boolean function $\mathsf{F} : \mathcal{F} \to \{0, 1\}$, where $\mathcal{F} \subseteq [M]^N$. For example, $\mathsf{F}(f) = 1$ if the input function f is one-to-one and $\mathsf{F}(f) = 0$ if f is two-to-one. Clearly, F can take binary oracles as input by restricting M to $\{0, 1\}$.

2.10 Classical Algorithmic Review

Here is a reminder of some classical algorithmic concepts, which may be helpful to understand their quantum counterparts. Before starting, recall that our goal is to compute some partial Boolean function $\mathsf{F} : [M]^N \to \{0, 1\}$ where the input is a finite tuple $x = x_1 x_2 \ldots x_N$ given as a black-box. The material of this section is mainly based on references [22, 31, 78]. Throughout this section, we consider algorithms in the *random access model.* We will assume analogous quantum model, in particular the quantum random access memory (see Section 2.11.3).

2.10.1 Bounded-Error Probabilistic Algorithm

Most of problems have unknown efficient algorithms (deterministic or probabilistic) that are able to return the correct and exact solution every time. Here *probabilistic* means that an algorithm is able to make random choices (coin flips) that hopefully guide it to the correct solution more quickly. Probabilistic or randomized algorithms might be divided into two categories: bounded-error (*Monte Carlo*) algorithms and zero-error (*Las Vegas*) algorithms. We deal with the first category.

While the answer returned by a deterministic algorithm is always correct, if no hardware error occurs, a *Monte Carlo* algorithm occasionally makes mistakes, but it finds solutions with high probability whatever the instance under consideration. There must be no instance on which the probability of error is high. A Monte Carlo Algorithm $\mathcal{A}$ is called *p-correct*, for $0 < p < 1$, if it returns the correct answer with probability at least p whatever the instance considered [22].

An algorithm $\mathcal{A}$ computes F with *bounded-error* ε if its output equals $\mathsf{F}(x)$ with probability at least $p = 1 - \varepsilon$ for every x in the domain. The *complexity* of F is the minimum integer T over all $\mathcal{A}$ computing F in time T.

A Monte Carlo algorithm may be *one-sided* or *two-sided* errors. For decision problems, one-sided algorithms are classified as either *false-biased* or *true-biased*. A true-biased algorithm is always correct when it returns true; a false-biased behaves likewise. Two-sided errors algorithm has no bias: the answer (either true or false) will be incorrect, or correct, with some bounded probability. This property (biased algorithms) could be useful to reduce the error probability arbitrarily at the cost of slight increase in running time (see Section 2.10.2).

For example, Grover's algorithm always returns 0 when there is no x such that $\mathsf{F}(x) = 1$ whereas it returns 1 with probability better than $(1 - 1/N)$ otherwise. Simply put, "true" answers from the algorithm are certain to be correct whereas "false" answers remain uncertain. This is said to be a $(1-1/N)$-correct true-biased algorithm for unstructured search problems.

A randomized algorithm has *worst-case success probability* p if, for every problem instance, the algorithm returns a correct answer with probability at least p.

A randomized algorithm has *average-case success probability* p if, averaged over all problem instances of a specified size, the probability that the algorithm returns a correct answer is at least p [74].

2.10.2 Probability Amplification

Monte Carlo algorithms are *biased*, which is a property that allows us to reduce the error probability (for decision problems). This technique was called *"amplification of the stochastic advantage"* [22]. However, we called it *probability amplification* to make analogy with the quantum "amplitude amplification" (Section 2.12.3).

For a one-sided error algorithm, the failure probability can be reduced arbitrary at slight increase of computing complexity by running the algorithm k times, thus amplifying the success probability up to a desired constant that depends on k.

For a two-sided error and p-correct algorithm, provided that $p > 1/2$, the failure probability may be reduced by running the algorithm k times and returning the majority function of the answers.

2.10.3 Probabilistic Query Algorithm

A query algorithm computing a function F is one whose input x is given as an oracle. A *query* typically sends an index i at a time and receives the element x_i. An algorithm computing in this model is adaptive, meaning that the kth query depends on its total history (answers to $k-1$ previous queries). A query algorithm, which is measured in query complexity, may be *deterministic* or *probabilistic* (randomized). The *query complexity* of a function F is the minimum integer T over all algorithms computing F with T queries.

A *deterministic algorithm* is one whose input deterministically controls its output. We say that it computes F if its output equals $\mathsf{F}(x)$ for every x in the domain. The complexity of F is taken over all algorithms computing F *exactly*.

A *probabilistic (query) algorithm* uses (typically uniformly-distributed) random bits as a guide to its behaviour in the hope of achieving a better performance than the deterministic one. An input x no longer determines the algorithm result with certainty, which now becomes 0 or 1 with a certain probability. A randomized algorithm computes F with bounded-error probability at most ε whenever its output equals $\mathsf{F}(x)$ with probability *at least* $1-\varepsilon$ for every input x in the domain.

2.11 Quantum Algorithmic Review

Based on several references [10, 31, 78], we specify the framework of quantum algorithms, which differ form their classical counterparts. For quantum computing in general, the reader is referred to textbooks of Kaye and Laflamme and Mosca [51], and Nakahara and Ohmi [66]. As in the classical framework, our goal is to compute some partial Boolean function $\mathsf{F} : R^N \to \{0, 1\}$ where the input is a finite N-tuple $x = x_1 x_2 \dots x_N$ given as a black-box. We take $R = \{0, 1\}$ just for convenience.

2.11.1 Quantum Query Algorithm

A quantum query algorithm $\mathcal{A}$ that makes T queries is the quantum analogue to a classical query algorithm with T queries, but now queries can be in *superposition.*

A T-query quantum algorithm $\mathcal{A}$ starts with the *all-zero* state $|0 \cdots 0\rangle$, evolves by applying a sequence of arbitrary unitary transformations $U_0, \dots, U_T$, alternated with T queries O_x to the oracle x, followed by a measurement of the final state, producing some *classical outcome* as a result of the computation. More formally, for any $T \geqslant 0$ and an oracle x, the final state of a T-query $\mathcal{A}$ is denoted by

$$|\Phi_x^T\rangle = U_T O_x U_{T-1} O_x \dots U_1 O_x U_0 |0 \cdots 0\rangle,$$

which is then measured. The U_i's are arbitrary unitary transformations that do *not* depend on the input x while the oracle unitary transformations O_x are all equal and depend on x.

To compute a function F, a quantum computer mainly uses three registers $|i\rangle|a\rangle|z\rangle$. The register $|i\rangle$ is for the query *index*, the register $|a\rangle$ holds the query *answer*, and $|z\rangle$ denotes an arbitrary fixed number of *working* qubits. The algorithm $\mathcal{A}$ is working in the vector space $\mathcal{H}$ spanned by the basis vectors $\{|i\rangle|a\rangle|z\rangle\}$. Assuming that m is the total number of qubits, each transformation acts on the m qubits and there are 2^m basis states for each stage of computation. For convenience, the basis states are usually specified using natural numbers $|0\rangle, |1\rangle, \cdots, |2^m - 1\rangle$

corresponding to their binary representations. Notice that $|z\rangle$ may consists of two parts: qubits for *reversible computation* that are returned to their original state at the end of the computation, and qubits holding the *result of computation*.

Any state $|\Phi\rangle$ can be uniquely written as $|\Phi\rangle = \Sigma_k \alpha_k |k\rangle$ where k varies over all basis states and the α_k's are complex numbers such that $\Sigma_k |\alpha_k|^2 = 1$. Measuring $|\Phi\rangle$ in the above basis produces a *classical result* k with probability $|\alpha_k|^2$. The measurement operation is not reversible as opposed to unitary transformations.

Considering a binary oracle x, the index i has length $\lceil \log N \rceil$ bits and the answer a is one bit. By convention, the rightmost bit of the final state $|\Phi_x^T\rangle$ denotes the output of computation after measurement. For computing non-Boolean functions, the reader is referred to Ref. [2].

An algorithm $\mathcal{A}$ computes F with bounded-error probability at most ε if its result equals $\mathsf{F}(x)$ with probability *at least* $1 - \varepsilon$ for every input x. The *query complexity* of F is the minimum integer T over all (bounded-error) algorithms computing F with T queries and error ε. Accordingly, the transformations U_i's are costless in terms of query complexity since they do not depend on the oracle.

2.11.2 Quantum Query Implementation

There are two natural ways of modelling a reversible query to an arbitrary Boolean function $f : \{0,1\}^N \to \{0,1\}$. The first way is

$$O_x |i, a, z\rangle \longrightarrow |i, a \oplus x_i, z\rangle$$

where $\oplus$ denotes the bitwise exclusive-or. Recall that $|i\rangle$ is an n-qubit register while a is a 1-qubit register. We can extend this definition to allow a *non-query* which can be obtained by setting either $i = 0$ or $x_i = 0$.

The second way to implement a query is

$$O_x |i, z\rangle \longrightarrow (-1)^{x_i} |i, z\rangle$$

and we usually say that the oracle is "computed in the phases" by O_x. It is very simple to invert the amplitude of exactly those states with $f(i) = 1$.

The two models are equivalent, but one may be more convenient depending on the context. For instance, the second one is more convenient to prove lower bounds [78]. For completeness, there is a reformulation of the first quantum query, which is the key idea to establish the relation between quantum algorithms and degrees of polynomials, allowing to prove lower bounds using the polynomial method [10]. Formally, the third query implementation can be written as

$$O_x|i, a\rangle \longrightarrow (1 - x_i)|i, a\rangle + x_i|i, a \oplus 1\rangle.$$

2.11.3 Quantum Random Access Memory

Nowadays, a quantum computer is imagined to be analogous to the successful classical architecture, in the sense that it consists of two fundamental components: a central processing unit (CPU) and a quantum random access memory (qRAM). A simple qRAM may contain 2^n memory cells (classical or quantum). Each cell is uniquely identified by an integer $0 \leqslant x \leqslant 2^n - 1$, which is then called the address. To access a cell x, its address is loaded into the address (or index) register, then the content of the corresponding memory cell c_x is provided in the data register. Unlike a classical memory, if the index register is in a *superposition of addresses*, a qRAM provides a superposition of pairs: (*address, correlated data*). We explain differently. Let $|0\rangle_a$ and $|0\rangle_d$ denote, respectively, the address and data registers, set to zero initially. If the input in a superposition of states $\sum_{x=0}^{2^n-1} \alpha_x |x\rangle_a |0\rangle_d$, we obtain the entangled output state $\sum_{x=0}^{2^n-1} \alpha_x |x\rangle_a |c_x\rangle_d$. Therefore, with one addressing process we have access to an exponential amount of data, contrasting with the classical memory that returns the content of only one memory cell at a time.

Such a memory is the main factor of improvements (in query complexity) of several important quantum algorithms [4, 20, 25] over their classical counterparts. Actually, these algorithms presume that this huge amount of memory contents can be loaded in at most linear time.

Unfortunately, there are still technological issues regarding the implementation of such architecture. More precisely, in a classical memory, one addressing process allows to load the content of a single memory cell, activating only a linear number of gates (that determine a single path). However, in a quantum computer, an address register in a superposition entails activating an exponential number of gates (the all possible paths), creating amongst others the *coherence* problem, which is a serious technological challenge for quantum information processing. In fact, even theoretical proposals are still unclear if they really solve the problem. This topic is beyond the scope of our research. Readers are referred to papers of Giovannetti, Lloyd and Maccone [43] or the recent paper of Hong, Xiang, Zhu, Jiang and Wu [47].

For this theoretical thesis, we assume that a qRAM as we defined is available. In particular, we take into account the query complexity only.

2.12 Quantum Search Tools

A large number of problems can be reduced to search problems of the form "find some value v in a set of possible inputs such that the statement $f(v)$ is true". Such problems includes database search, sorting and attacking an encryption scheme.

For example, consider a cryptographic scenario where an eavesdropper has intercepted matching pairs of plaintext and ciphertexts, and the goal is to find the key that maps one into the other. This problem can be easily viewed as a search for the secret key k for which the statement "k can decrypt all the given ciphertexts" is true. This section is dedicated to briefly review fundamental quantum search methods for structured and unstructured search problems.

An *unstructured* search problem is one where nothing is known about the structure of the search space or the statement $f(v)$. Randomly testing the truth of $f(v)$ one by one is the optimal method. For instance, if f is a black-box function, then inverting $f(v)$ is an unstructured search problem.

However, in a *structured* search problem (e.g., searching an alphabetized list), information about the search space or f can be exploited to provide faster search [69].

2.12.1 Grover's Algorithm

Consider an oracle function $f : D \to R$ and an image $y \in R$ with the promise that there exists one and *only one* $x \in D$ such that $f(x) = y$. The problem is to find x, which is then called a *solution*.

Classically (deterministically or probabilistically), there is no better strategy than trying distinct inputs at random until an x is found such that $f(x) = y$. This requires trying $N/2$ inputs on the average and $N - 1$ on the worst case. However, Grover's algorithm [46] solves this problem in

$$O(\sqrt{N})$$

quantum queries to f with bounded-error probability $1/N$. Actually, this error probability can be reduced to *zero* [24]. Besides, Grover's algorithm is not only optimal [19, 20] but exactly optimal [39]. Therefore, it is provably more efficient than any algorithm running on a classical computer for problems that can be modelled as a black-box. Readers interested in more description and analysis of this algorithm are refereed to references [20, 51, 55, 69].

2.12.2 Generalized Grover's Algorithm

Here we review the algorithm for the generalized unstructured search problem. Let $f : D \to R$ be a black-box function and $Y \subset R$ be such that $|Y| = t$ with $t \ll |R|$. The problem is to find any x such that $f(x) \in Y$, which is then called a *solution*.

Whether or not t is known, Boyer, Brassard, Høyer, and Tapp [20] solved this problem by generalizing Grover's algorithm. We are interested in problems where the number of solutions is known; for convenience we refer to this algorithm as BBHT's algorithm (generalization) or simply BBHT.

All remain similar to Grover's algorithm except for few necessary modifications. Let t be the number of solutions, assumed to be known, and let θ be such that

$\sin^2\theta = t/N$ with $|D| = N$. Generalized Grover's search provides a solution in

$$O(\sqrt{N/t})$$

quantum evaluations of f with bounded-error probability at most t/N. This error probability is vanishing whenever $t \ll N$, which is usually the case in practice. Note that a variant of this algorithm allows us to find the correct answer with *certainty* [24].

2.12.3 Amplitude Amplification

Consider a Boolean function $f : X \rightarrow \{0, 1\}$ that partitions a set X into two sets of *good* and *bad* elements, where x is good if $f(x) = 1$ and bad otherwise. Assume also that a quantum algorithm $\mathcal{A}$ such that $\mathcal{A}|0\rangle = \Sigma_{x\in X}\alpha_x|x\rangle$ is a quantum superposition of all elements of X. Let p denote the probability that a good element is produced if $\mathcal{A}|0\rangle$ is measured.

If we repeat the process of running $\mathcal{A}$, measuring the output, and using f to check the validity of the result, we shall expect to repeat $1/p$ times on the average before a solution is found. However, assuming algorithm $\mathcal{A}$ makes no measurements, amplitude amplification is a process that allows to find a good x after an expected number of applications of $\mathcal{A}$, and its inverse $\mathcal{A}^{-1}$ in $O(1/\sqrt{p})$. This process works whether or not the value of p is known ahead of time. The value of p in the problems we consider is always known. Formally, we state the following theorem due to Brassard, Høyer, Mosca, and Tapp [26].

Theorem 2.12.1 (Quadratic speedup with known p)**.** *Let $\mathcal{A}$ be any quantum algorithm that uses no measurements, and let $f : Z \rightarrow \{0, 1\}$ be any Boolean function. Given the initial success probability $p > 0$ of $\mathcal{A}$, there exists a quantum algorithm that finds a good solution with certainty using a number of applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$ which is in $\Theta(1/\sqrt{p})$ in the worst case.*

2.12.4 Quantum Walks on Johnson Graphs

Our optimal quantum attacks are modelled as quantum walks on different graphs. We review in this section quantum walks on Johnson graphs, which was devised for element distinctness by Ambainis [4], and in Section 2.12.5 the general framework. Most of quantum walks material are closely based on Santha's excellent survey [70].

A Johnson graph $J(N, r)$ is an undirected graph whose vertices are the r-subsets (of distinct elements) of $[N]$ and there is an edge between two nodes if and only if they differ by exactly one element. Intuitively, we may think of "walking" from one node to an adjacent node by dropping one element and replacing it by another. The task is to find a specific k-subset of $[N]$. The nodes that contain this subset are called *marked*.

A random walk P on a Johnson graph can be quantized and the cost of the resulting quantum algorithm can be written as a function of S, U and C. These are the costs of preparing the state related to the stationary distribution (*setup* phase), moving unitarily from one vertex to an adjacent vertex (*update* phase) defined by the chain, and checking whether a vertex is marked (*checking* phase), respectively.

Theorem 2.12.2. *[4] Let M be either empty, or the set of vertices that contain a fixed subset of constant size $k \leqslant r$. Then there is a quantum algorithm that* finds, *with high probability, the k-subset if M is not empty at an expected cost in the order of*

$$\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right),$$

where $\delta \in \Theta(\frac{1}{r})$ is the eigenvalue gap of the symmetric walk on $J(N, r)$ and $\varepsilon \in \Omega(\frac{r^k}{N^k})$ is the probability that a random node is marked.

Childs and Eisenberg [36] proved that this algorithm can be used beyond element distinctness, and provided a much simpler analysis than that of Ambainis. It can be used in any application that can be reduced to the *subset-finding* problem.

2.12.5 Quantum Walks on Markov Chains

Szegedy [75] then Magniez, Nayak, Roland and Santha (MNRS) [59] gave a general framework to derive quantum search algorithms from a large class of Markov chains.

Given a Markov chain P on a discrete space X, with $|X| = N$, and a subset of marked elements $M \subseteq X$, the problem is to provide an upper bound on the number of iterations to encounter an element from M for the first time. We identify a Markov chain over state space X with its transition matrix $P = (p_{xy})$ where p_{xy} is the probability of transition from state x to state y.

Random walks on a directed weighted graph $G(V, E)$ can be modelled by a Markov chain defined by the sequence of moves taken by a robot between vertices of G. The vertices are the states of the chain, the place of the robot at a given time step is the state of the process, and there is an edge $(x, y) \in E$ if and only if $p_{xy} > 0$, in which case the weight of the edge (x, y) is defined by p_{xy}. In a d-regular graphs, the probability that the robot follows edge (x, y) is $1/d$.

Thanks to Szegedy's theorem [75], a random walk P on some graphs can be quantized and the cost of the resulting quantum algorithm can be written as a function of S, U and C defined in the previous section. Szegedy's algorithm [75] is the quantum analogue for the class of ergodic and symmetric Markov chains. More precisely, it provides a framework for "constructing and characterizing the behaviour of a quantum walk algorithm by specifying a classical random walk, and analyzing its spectral gap and stationary distribution [58]".

Theorem 2.12.3 (Szegedy [75])**.** *Let P be an ergodic and symmetric Markov chain, and let ε be a lower bound on $\frac{|M|}{|X|}$ whenever M is non-empty. Then there is a quantum algorithm that* determines *with high probability if M is non-empty at a cost in the order of*

$$\mathsf{S} + \frac{1}{\sqrt{\delta\varepsilon}}\Big(\mathsf{U} + \mathsf{C}\Big),$$

where δ is the eigenvalue gap of the chain P.

Based on Szegedy's result [75], Magniez, Nayak, Roland and Santha provided a quantum algorithm (MNRS) [59] for ergodic and reversible Markov chains.

Theorem 2.12.4 (MNRS [59])**.** *Let P be an ergodic and reversible Markov chain, and let $\varepsilon > 0$ be a lower bound on the probability that an element chosen from the stationary distribution of P is marked whenever M is non-empty. Then, there exists a quantum algorithm that* finds, *with high probability, an element of M, if there is any, at a cost in the of order*

$$\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right).$$

2.12.6 Random Walks for Testing Group Commutativity

Some of our cryptanalysis algorithms are modelled as quantum walks, inspired by Magniez and Nayak's algorithm for testing group commutativity [58], also analysed using Szegedy's or MNRS's theorem [59, 75].

Consider a set X of N elements. The random walk takes place on an undirected graph $G(V, E)$ whose vertices are the r-tuples of distinct elements of X. Two vertices $u = (u_1, \ldots, u_r)$ and $v = (v_1, \ldots, v_r)$ are connected if and only if v is obtained from u by permuting two of its *components* (not necessarily distinct); or replacing one of its components by an element from X (not in the node). The stationary distribution is the uniform distribution since the graph is symmetric. The eigenvalue gap $\delta \in \Omega(1/(r \log r))$ while the probability $\varepsilon \in \Omega(r^2/N^2)$. From a vertex $u = (u_1, \ldots, u_r)$, we define the following transitions:

– with probability $1/2$, stay at u,

– with probability $1/2$, pick $i \in [r]$ and $j \in X$; if $j = u_k$ for some $k \in [r]$, then exchange u_i and u_k; otherwise, set $u_i = j$.

2.12.7 Random Walks on Hamming Graphs

We will also provide quantum attacks based on quantum walks on Hamming graphs due to Childs and Kothari [35] (see Section 6.3). A hamming graph $H(X, r)$ has vertex set X^r and there is an edge between two r-tuples if and only if they differ only on *one* coordinate. The eigenvalue gap of this random walk is $\delta = \Omega(1/r)$.

2.13 Lower Bound Methods

The most successful techniques for proving lower bounds on quantum query complexity are the polynomial method [10], the adversary method [3] and the generalized adversary method [81].

The polynomial method was given by Beals, Buhrman, Cleve, Mosca and de Wolf [10]. The main idea is that any quantum algorithm computing some Boolean function F produces a corresponding polynomial. In particular, if the algorithm computes F with probability at least $1-\varepsilon$, the polynomial approximates the function F to within ε at all points in the domain. Therefore, by proving a lower bound on the degree of polynomials approximating F, we can derive a lower bound on the number of queries the quantum algorithm needs to make.

The first quantum lower bound was given by Bennett, Bernstein, Brassard, and Vazirani [19], which is known as the *hybrid* method. Using classical adversary argument, this method and all its subsequent variants are based on one principle: we transform the problem to *distiguishing* inputs instead of computing the function. Consider an algorithm $\mathcal{A}$ that computes successfully some Boolean function F in the oracle model, and two inputs x, y such that $\mathsf{F}(x) \neq \mathsf{F}(y)$. Then $\mathcal{A}$ must be able to *distinguish* between oracles x and y. More precisely, for a given problem an adversary runs $\mathcal{A}$ on one input, then changes the input slightly so that the function value changes but the algorithm cannot recognize this change unless it makes a large number of queries. Therefore, deriving a good lower bound is reduced to identify two inputs that are hard to distinguish.

The measure of distinguishability is the inner product. Initially, the inner product is one since the computation starts in a fixed state, and the output quantum states of $\mathcal{A}$ on x and y must be almost orthogonal. Therefore, upper-bounding the change of the inner product (the reduced uncertainty) after a single query implies a lower bound on the required number of queries.

Ambainis [3] generalized this method, which is known today as the *unweighted* adversary method. It became a very successful technique, having provided tight lower bounds for several interesting problems. It starts with choosing a set of pairs, as opposed to one single pair in the first method, that maps F to different values. Then the lower bound is brought to some combinatorial properties of these pairs.

However, considering only the hardest inputs does not produce good lower bounds for some problems, such as sorting and searching an ordered list for instance. Høyer, Neerbek and Shi introduced the weighted adversary arguments to prove tight bounds for these problems [79]. The idea is to assign weights that represent the hardness (in terms of queries) of distinguishing each pair of inputs. For this purpose, one defines the spectral adversary matrix.

Definition 2.13.1. *A* spectral adversary matrix *for a fixed function* $\mathsf{F} : S \to T$ *is a real symmetric matrix* $\Gamma : S \times S \to \mathbb{R}$ *such that* $\Gamma[x, y] = 0$ *whenever* $\mathsf{F}(x) = \mathsf{F}(y)$.

There are different means to determine the progress an algorithm makes in order to distinguish inputs, which led to formulations in terms of weight schemes due to Ambainis and Zhang [7, 82], Kolmogorov complexity due to Laplante and Magniez [54] and in terms of eigenvalues due to Barnum, Saks and Szegedy [9]. All these formulations are based on the same technique: bound the difficulty of distinguishing inputs. Using the duality theory of semidefinite programming, Špalek and Szegedy [76] showed that all these formulations are equivalent.

The adversary method and all its versions have several inherent limitations. One limitation in our context is that they are incapable to derive good lower bounds for any problem subject to the *certificate complexity barrier.* This limitation states that $\mathsf{ADV}(\mathsf{F}) \leqslant \sqrt{C_0(\mathsf{F})C_1(\mathsf{F})}$ for total functions [76, 82], where $C_b(F)$ is the b-certificate complexity of F. However, $\mathsf{ADV}(\mathsf{F}) \leqslant 2\sqrt{C_1(\mathsf{F})N}$ if the function is partial [76], where N is the input size. The same shortcoming for the Kolmogorov complexity method was proved by Laplante and Magniez [54]. Consequently, for a problem like element distinctness, where one of the certificate complexities is constant, the best bound which can be proven by the adversary method is $\Omega(\sqrt{N})$. This contrasts with the polynomial method, which enabled to prove a tight lower bound of $\Omega(N^{2/3})$ for this problem [2].

Informally, certificate complexity measures how many of the N boolean variables should be given values so that the function's value is fixed.

Definition 2.13.2 (Certificate complexity)**.** *Let $f : \{0,1\}^N \to \{0,1\}$ be a function and $C : S \to \{0,1\}$ be an assignment of values to some subset S of the N indices. We say that C is consistent with $x \in \{0,1\}^N$ whenever $x_i = C(i)$ for all $i \in S$. A* 1*-certificate for f is an assignment C such that $f(x) = 1$ whenever x is consistent with C. The size of C is the cardinality of S. We similarly define a* 0*-certificate. The certificate complexity $C_x(f)$ of f on x is the size of the smallest $f(x)$-certificate that is consistent with x. The certificate complexity $C(f)$ of f is the maximum of $C_x(f)$ over all x. The* 1*-certificate complexity of f is the maximum of $C_x(f)$ over all x for which $f(x) = 1$.*

For example, the certificate complexity of the **OR** function on $(1, \ldots, 0)$ is 1, because the assignment $x_0 = 1$ forces the **OR** to 1. The same holds for the other x for which $\mathbf{OR}(x) = 1$, so the 1-certificate is 1. On the other hand, the certificate complexity on $(0, \ldots, 0)$ is N. Therefore, $C(\mathbf{OR}) = N$.

A stronger version of the adversary method was given by Høyer, Lee and Špalek. Called negative (or generalized) adversary method, it is also known as $\mathsf{ADV}^{\pm}$ [81]. This new method is always at least as good as the adversary method and can break the certificate complexity barrier. Indeed, there is a monotone function f for which $\mathsf{ADV}^{\pm}(f) = \Omega(\mathsf{ADV}(f)^{1.098})$ [81] and Belovs [14] constructed an explicit optimal (negative-weight) adversary matrix for element distinctness. On top of that, the generalized adversary bound is *tight* for partial and total functions as shown by Lee, Mittal, Reichardt, Špalek and Szegedy showed [56]. We will see in Chapter 4 how these results [56, 81] enable us to provide a security proof of our protocols, after trying in vain several lower-bound techniques. Moreover, the new method $\mathsf{ADV}^{\pm}$ has all the advantages of the adversary method ADV, particularly: it is a lower bound on the bounded-error quantum query complexity, and has a very useful property with respect to *function composition*, which is actually the corner stone of all our lower bound proofs.

We now present a closer look at this new method and compare it with previous adversary methods, using the setting of the spectral formulation of the adversary method [9]. Before proceeding, following Refs. [80, 81], we recall briefly this spectral formulation.

Let $Q_2(\mathsf{F})$ denote the two-sided bounded-error quantum query complexity of a function F. For a real matrix M we use $M \geqslant 0$ to say its entries are nonnegative, and $\|A\|$ denotes the *spectral norm* of A (which is equal to its largest eigenvalue). In the following, D_i is the zero-one valued matrix defined by $D_i[x, y] = 1$ if and only if the bitsrings x and y differ in the i-th bit or, equivalently, $D_i[x, y] = 1$ if $x_i \neq y_i$. For two matrices A, B of the same size, the entrywise (or Hadamard) product is the matrix denoted by $A \circ B$ and defined by $(A \circ B)[x, y] = A[x, y]B[x, y]$.

Suppose we want to determine the quantum query complexity of a function F. First, we assign weights to pairs of inputs in order to bring out how hard it is (in terms of number of queries) to distinguish these inputs apart from one another. The adversary lower bound is the worst ratio of the spectral norm of this matrix, which measures the overall progress necessary in order for the algorithm to be correct, to the spectral norms of associated matrices, which measure the maximum amount of progress that can be achieved by making a single query.

The spectral adversary method states that, for any F, the bounded-error query complexity $Q_2(\mathsf{F})$ is lower-bounded by a quantity $\mathsf{ADV}(\mathsf{F})$ defined in terms of Γ.

Definition 2.13.3. *Let Γ be an adversary matrix for a fixed function $\mathsf{F} : S \to T$. The adversary bound of F using Γ is*

$$\mathsf{ADV}(\mathsf{F}; \Gamma) = \min_i \frac{\|\Gamma\|}{\|\Gamma \circ D_i\|}.$$

The adversary bound of F is

$$\mathsf{ADV}(\mathsf{F}) = \max_{\substack{\Gamma \geqslant 0 \\ \Gamma \neq 0}} \mathsf{ADV}(\mathsf{F}; \Gamma).$$

Theorem 2.13.4 (BSS03 [9])**.** *For any function F, $Q_2(\mathsf{F}) = \Omega(\mathsf{ADV}(\mathsf{F}))$.*

Definition 2.13.5. *Let Γ be an adversary matrix for a fixed function $\mathsf{F} : S \to T$. The adversary bound of F using Γ is*

$$\mathsf{ADV}^{\pm}(\mathsf{F};\Gamma) = \min_i \frac{\|\Gamma\|}{\|\Gamma \circ D_i\|}.$$

The adversary bound of F is

$$\mathsf{ADV}^{\pm}(\mathsf{F}) = \max_{\Gamma \neq 0} \mathsf{ADV}^{\pm}(\mathsf{F};\Gamma).$$

Theorem 2.13.6 (HLS07 [81])**.** *For any function F, $Q_2(\mathsf{F}) = \Omega(\mathsf{ADV}^{\pm}(\mathsf{F}))$.*

We can see from the definitions that $\mathsf{ADV}^{\pm}(\mathsf{F}) \geqslant \mathsf{ADV}(\mathsf{F})$ for any function F. Indeed, while the definition of ADV restricts the maximization to matrices Γ whose entries are nonnegative and real, the new bound $\mathsf{ADV}^{\pm}$ removes this restriction.

The key idea to transmit is that finding a good lower bound is simply reduced to picking a good adversary matrix Γ. However, finding such a good Γ is not obvious.

2.13.1 New Lower Bound Composition Theorem

The central technical part of our lower bounds consists in analyzing the complexity of a function closely related to the hardness of breaking our key agreement protocols. Recall that X' denotes $X \cup \{0\}$, where X is an arbitrary set of integers. This function is obtained when composing a general problem (*subset-finding*) with κ instances of a variant of the unstructured search problem.

Consider three integer parameters κ, η and k, and three functions $f : [\kappa] \to [\kappa]$, $g : [\kappa] \to [\eta]$ and $h : [\kappa] \times [\eta] \to [\kappa]'$ so that

$$h(i,j) = \begin{cases} f(i) & \text{if } j = g(i), \\ 0 & \text{otherwise}. \end{cases}$$

The task is to find a unique k-subset of distinct non-zero elements which satisfy some property (equality for instance), having access to a black-box function h only.

More precisely, find non-zero elements $i_1, \ldots, i_k \in [\kappa]$ that would provide a solution to the outer poblem if we were given direct access to f. This can be viewed as *searching* among η possibilities for the sole nonzero $h(i, \cdot)$ for each i and then solving f on those *elements*.

It is more convenient to prove this lower bound for the related decision problem: we are given a function h of the type above, but it is based on a function f that either has a single k-subset, or none. The task is to decide which is the case. Obviously, any algorithm that can solve the search problem with probability of success at least $p > 0$ can be used to solve the decision problem with error bounded by $\frac{1}{2} - \frac{p}{2}$: run the search algorithm; if a k-tuple is found (and verified), output "yes", otherwise output either "yes" or "no" with equal probability after flipping a fair coin. It follows that any lower bound on the bounded-error decision problem applies equally well to the search problem.

We change the notation to adapt it to the standard usage in the field of quantum query complexity. The function $f : [\kappa] \to [\kappa]$ is represented by an element of $[\kappa]^\kappa$. This makes it possible to think of the decision version of this problem as a Boolean function $\mathsf{F} : [\kappa]^\kappa \to \{0, 1\}$. Given κ integers $(z_1, \ldots, z_\kappa) \in [\kappa]^\kappa$, or equivalently on input f, the goal is to decide whether or not there is a k-subset providing a solution to f by making as few queries as possible.

We compose F with κ instances of a promise version of a search problem, which we call $\mathsf{pSEARCH}$.

Definition 2.13.7. $\mathsf{pSEARCH} : P \to A$ *with* $P \subseteq (A')^\eta$ *is a promise problem. On input* $(a_1, \ldots, a_\eta)$*, the promise* P *is that all but one of the values are zero. The goal is to find and output this nonzero value by making queries that take* i *as input and return* a_i*.*

The composed function, with $A = [\kappa]$, is denoted H. On input $x \in P^\kappa$,

$$\mathsf{H}(x) = \mathsf{F}(\mathsf{pSEARCH}(x_1), \ldots, \mathsf{pSEARCH}(x_\kappa)).$$

We can think of H as a two-level tree: the root being labelled by F and each of the κ leaves being labeled by $\mathsf{pSEARCH}$. An input x to H can be thought of as being comprised of κ parts, $x = (x_1, \ldots, x_\kappa)$. We evaluate H on x by first finding the κ non-zero values, and then computing F on these κ values.

Since H is defined as the composition of F and $\mathsf{pSEARCH}$, we would like to apply a new composition theorem for the generalized adversary method [81], which would say that if a function $\mathsf{H} = \mathsf{F} \circ \mathsf{G}^\kappa$, then $\mathsf{ADV}^\pm(\mathsf{H}) \geqslant \mathsf{ADV}^\pm(\mathsf{F}) \cdot \mathsf{ADV}^\pm(\mathsf{G})$. Unfortunately, the composition theorems of Refs. [56, 81] require the inner (and outer [81]) functions to be Boolean, which is not the case here for the inner function $\mathsf{pSEARCH}$. Since counter-examples can be found, we cannot hope to have a fully general composition theorem in which the inner function would be an arbitrary function. Nevertheless, we proved a new composition theorem with $\mathsf{pSEARCH}$ as the inner function [29].

Theorem 2.13.8 (BHKKLS11 [29])**.** *Let* $\mathsf{F} : A^\kappa \to B$*,* $\mathsf{pSEARCH} : P \to A$ *with* $P \subseteq (A')^\eta$ *as described above, and* $\mathsf{H} = \mathsf{F} \circ \mathsf{pSEARCH}^\kappa$*. Then*

$$\mathsf{ADV}^\pm(\mathsf{H}) \geqslant \frac{2}{\pi} \mathsf{ADV}^\pm(\mathsf{F}) \cdot \mathsf{ADV}^\pm(\mathsf{pSEARCH}).$$

The quantum query complexity of H is in $\Omega(\kappa^c \eta^{1/2})$ where c is a positive constant.

Now the necessary condition on F can be easily seen. Indeed, the adversary bounds of each of the involved functions are needed before invoking the theorem.

2.14 Proving Cryptographic Lower Bounds

While we always assume that security is maintained only against linear query Eave, which has full access to any communicated message over a public authenticated channel, we make *no assumption* on Eave's strategy. Therefore, to prove that a given protocol Π is secure requires proving a lower bound on the number of queries needed by any eavesdropper attempting to break Π. In other word, we have to prove that no linear query algorithm can break Π, except with vanishing probability.

However, the current state of complexity theory does not allow to solve such cryptographic problem straightforwardly. The alternative strategy is to first prove a lower bound on the difficulty to solve a related problem X, then prove that the scheme Π is secure as long as X preserve the proven difficulty. This strategy proceeds by devising a reduction (an argument by contradiction) which converts any probabilistic linear query algorithm $\mathcal{A}$ able to break Π (with non-vanishing probability) into an algorithm $\mathcal{A}'$ able to solve the problem X more quickly than what was proven. For a detailed application of this strategy, see Section 4.2.2. Actually, we use this approach to prove the security of all our protocols.

For more detail about this approach can be found in the textbooks [44, 60, 74] or papers [48, 49].

CHAPTER 3

MERKLE SCHEMES

In 1974, Merkle proposed the first solution [63] for the key agreement problem, where the third party is not needed, and the notion of public-key cryptography [38] as a project proposal in a graduate course on Computer Security (CS244) at the University of California, Berkely.

He proved that the key pre-agreement requirement is unnecessary, by a method which allows legitimate parties to establish a secret over a *public authenticated classical* channel after a number of queries proportional to some parameter N, while any classical eavesdropper needs a number of queries proportional to N^2 in order to obtain their secret from the communicated messages. The only assumption is the existence of "one-way encryption function" [63] of domain size N^2.

The proposal was rejected by the professor but Merkle "kept working on the idea". Initially rejected, it was eventually published in 1978 by Communications of the ACM [64]. Based on a concept called "puzzle", Merkle's published scheme [64] was different from his original scheme in the unclassified document [63], which is based on a variant of the *birthday problem* (see Section 2.8.6).

In the forthcoming sections, we describe Merkle's (original) scheme [63], its security analysis against a classical eavesdropper, and a quantum attack that makes it useless from a security viewpoint. Although unconsidered in this work, Merkle's published scheme [64] is described for completeness later. We finish the chapter by "Quantum Merkle Puzzles", which was introduced by Brassard and Salvail as a first attempt to recover Merkle's scheme which collapses in a quantum world.

While Merkle puzzles is known to mean Merkle's published scheme, unfortunately, it has been used [8, 29] to indicate the original one. In this thesis, we will consider only Merkle's original scheme [63], referring to it simply Merkle's scheme.

It is worth comparing these two schemes since they have several interesting differences: (1) at the time of writing, there are no lower bound methods in the literature to find the complexity of the published scheme; (2) it is an open problem whether this latter is optimal, in contrast with the original scheme [63], which is provably optimal in the black-box model [8]; and (3) there is no known reduction between them. On the other hand, these schemes also have similarities: (1) any classical eavesdropper needs an amount of queries which increases *quadratically* in the legitimate one before obtaining the secret; and (2) a quantum eavesdropper can find the secret as easy as the key agreement process (up to constant factors).

3.1 In a Classical World

All parties in this section are restricted to use *classical computers*, the 1970's world. We first describe Merkle's (original) scheme [63], afterwards the published one [64].

3.1.1 Merkle's Original Scheme

We emphasize that Merkle's unpublished scheme [63] is the only one we consider in this work and refer to it simply Merkle's scheme. Before the formal description, we first present the scheme according to Merkle's typewritten words [63].

```
Method:  Guessing.  Both sites guess at keywords.  These guesses
are one-way encrypted, and transmitted to the other site.  If
both sites should chance to guess at the same keyword, this
fact will be discovered when the encrypted versions are compared,
and this keyword will then be used to establish a communications
link.
Discussion:  No, I am not joking.  If the keyword space is of
size N, then the probability that both sites will guess at
a common keyword rapidly approaches one after the number of
guesses exceeds sqrt(N). Anyone listening in on the line must
examine all N possibilities.
```

More formally, assume the existence of a random oracle function $f : [N^2] \rightarrow [N^k]$, where the constant k is chosen large enough so that there is no collision in the image of f, except with vanishing probability (see Section 5.3.4).

The "keywords" guessed at by "both sites" are random distinct points in the domain of f. For more precision, let us describe the protocol at step $i > 2$, assuming it is Alice's turn to respond. Alice picks randomly a point $a_i \in [N^2]$ and transmits it encrypted, by the application of f, to Bob. Next, Bob picks a random point $b_j \in [N^2]$ and checks whether or not $f(b_i) \in Y_a$ where Y_a consists of the images sent from Alice. If the answer is 'yes', then Bob sends back $f(b_i)$ to Alice, who can learn b_i using elementary search in her table containing all her random points coupled with the corresponding images, and the value of b_i will be their secret. However, if the answer is 'no', then Bob proceeds exactly as Alice. Both sites continue comparing and transmitting encrypted random points to each other until they chance to guess on the same point.

Now, we ensure that the protocol is correct, that is, at the end of the execution of the protocol, Alice and Bob would have a common secret key after $O(N)$ queries. Indeed, since there are N^2 points in the domain of f, it is sufficient to do $O(N)$ random guesses at each site after which "both sites should chance to guess" at the same point, which becomes their common secret key. Note that the probability that both parties establish a secret after $O(N)$ queries approaches *one*, which can be proved using a simple probabilistic argument.

As for the security analysis, a classical eavesdropper who listens to the entire conversation has *no way* to obtain the secret key than to invert f on that common encrypted point, since f is given as a black-box. However, inverting f on any point in the domain requires an expected $\Omega(N^2)$ queries, since it requires trying on the average half the points in the domain.

It was a major open question in classical cryptography to determine whether this quadratic relation between the legitimate complexity and the eavesdropping one is the best possible in the black-box model. In 1989, Impagliazzo and Rudich showed that every key agreement protocol in the random oracle model in which

Alice and Bob ask $O(N)$ queries can be broken by an adversary asking $O(N^6 \log N)$ queries [49]. It took 35 years after Merkle's invention before Barak and Mahmoody-Ghidary proved that Merkle's scheme is indeed *optimal* in a classical world [8]. They designed an $O(N^2)$-query algorithm which can find the secret established by any key agreement protocol in the random oracle model in which Alice and Bob make $O(N)$ queries.

3.1.2 Merkle's Published Scheme

Eventually published in 1978 [64], this scheme has been known as *Merkle Puzzles*. Be aware that this name is also assigned to the original scheme [8, 29]. The idea is simple, and we describe it immediately following Ref. [23].

We asssume the existence of an *encryption function* f given as a black-box. Formally, let $f : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ where $\mathcal{K}$ is the encryption keyspace, $\mathcal{M}$ is the message space, and $\mathcal{C}$ is the ciphertext space. If k is a key and m is a message, then $c = f_k(m)$ is the ciphertext. Decryption is the inverse of f on c, given k. Symbolically, $m = f_k^{-1}(c)$. We assume that both f and f^{-1} can be computed in unit time provided k is available. However, guessing the key k, given c and arbitrary information about m, is only possible by exhaustive search on the keyspace. The size of $\mathcal{K}$ is assumed to be N in order to solve any puzzle in $O(N)$, but not faster. Assume also that Alice and Bob agree on an *arbitrary public value* v, which is used to verify whether or not a puzzle is solved, and should be of adequate size.

To *create a puzzle* P_i, for $1 \leqslant i \leqslant N$, choose randomly an encryption key k_i, a unique identifier ID_i, and a secret value x_i, then compute

$$P_i = f_{k_i}(ID_i, x_i, v).$$

However, to *solve* a puzzle P_i, the optimal way is to try random distinct keys $k \in \mathcal{K}$ until the finding of $f_k^{-1}(P_i) = (ID_i, x_i, v)$ for the right value of v, since f is a black-box encryption function. Note that identifying v is the only way to recognize that a puzzle is solved.

It is worth comparing with a cryptogram. A puzzle is simply a cryptogram which is meant to be solved efficiently (in linear number of queries here) while a cryptogram ideally cannot be decrypted efficiently. To achieve this objective (prepare solvable puzzles), it is sufficient to restrict the keyspace to an adequate size. The scheme might be described as follows:

1. Alice prepares, saves, and sends Bob N puzzles $P_1, P_2, \ldots, P_N$.
2. Bob selects *one* puzzle P_i for a randomly chosen $i \in [N]$, and solves it. The solution allows him to get ID_i and x_i. Bob transmits back to Alice the value of ID_i to inform her of the puzzle he has solved.
3. Alice, having received ID_i and using her saved puzzles, can know the puzzle that was solved by Bob using an elementary search. Hence, she can find efficiently the value x_i, which becomes their common secret key.

3.1.3 Advantages and Disadvantages of Merkle Schemes

Shortly after Merkle's original scheme [63], Diffie and Hellman [38] have discovered a method that makes the cryptanalytic computational complexity *apparently* exponentially harder than the legitimate complexity. However, there is no proof that the Diffie-Hellman public-key system is secure at all since it relies on the conjectured difficulty of extracting discrete logarithms, an assumption doomed to fail in a world ruled by the quantum theory. In contrast, Merkle's approach offers provable quadratic security against any possible *classical* attack in the black-box model, that is, f cannot be inverted by any other means than exhaustive search.

The major disadvantage of Merkle's scheme is that it provides no-better than polynomial security. Even worse, quadratic security is the best possible in a classical world [8]. In a quantum world, it is the objective of this thesis to know the extend to which we can push the security level, which is clarified in next chapters.

Next, we describe how Merkle's method collapses completely if the eavesdropper is equipped with a quantum computer, and review a partial solution to this problem [23] by granting similar powers to legitimate communicating parties.

3.2 In a Quantum World

From now on, we assume that the eavesdropper is *always* quantum, but legitimate parties might make use of quantum computing features depending on the context.

3.2.1 Collapse of Merkle's Scheme

While Merkle's scheme provides provable quadratic security in the random oracle model against any classical eavesdropper, it is broken in $O(N)$ queries against a quantum eavesdropper.

Indeed, here is a quantum attack applying directly Grover's algorithm [46]. Assume that $f(x)$ is the common encrypted point between Alice and Bob, and that the random oracle function f can be computed in a superposition of inputs. The eavesdropper would like to know the secret x.

The problem is reduced to an unstructured search problem where the search space is of size N^2 and there exists *one* and only one solution. The eavesdropping strategy is to resort to Grover's algorithm which can solve this problem after a number of queries proportional to the square root of the domain size, which is $O(N)$ quantum queries, and this is optimal [19, 20]. Grover's algorithm is reviewed in Section 2.12. The same attack can be used to collapse the published scheme too.

3.2.2 Quantum Merkle Puzzles

"Quantum Merkle Puzzles" were introduced in 2008 by Brassard and Salvail [23] as a first attempt to recover the security of Merkle's scheme.

Their protocol, in which a quantum eavesdropper has query complexity $\Theta(N^{3/2})$, is very similar to Merkle's scheme except for the following modifications: 1) allow Bob to make use of quantum computing; 2) increase the function domain size from N^2 to N^3; and 3) use the BBHT generalisation of Grover's algorithm.

1. Alice picks at random N distinct points, $x_1, x_2, \ldots, x_N$, in the domain of f and transmits them encrypted, by the application of the function f, to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$ and $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$.

2. Bob's task is to invert f on an arbitrary element in the set Y. He sorts the elements of Y so that he can quickly determine, given any y, whether $y \in Y$. Then, he defines the Boolean function $\phi : N^3 \to \{0,1\}$ as follows:

$$\phi(x) = \begin{cases} 1 & \text{if } f(x) \in Y \\ 0 & \text{otherwise} \end{cases}$$

   The problem is then reduced to finding x for which $\phi(x) = 1$. There are exactly N solutions satisfying this condition, out of N^3 points in the domain. Using BBHT, he can find a solution in $O(\sqrt{N^3/N}) = O(N)$ quantum queries.

3. Bob sends back $f(x)$ to Alice.

4. Alice, having kept her randomly chosen points, can efficiently find the value of x using elementary classical search. This x will be their secret key.

We calculate the legitimate complexity. Alice asks exactly N classical queries while Bob asks $O(N)$ quantum queries, neglecting the running time for sorting and binary search. Therefore, the legitimate query complexity is in $O(N)$.

The quantum eavesdropper, however, is faced to invert f on a specific point, which provably requires a number of queries proportional to the square root of the number of points in its domain [20, 24]. Therefore, the quantum eavesdropper needs

$$\Omega(\sqrt{N^3}\,) = \Omega(N^{3/2})$$

quantum queries, which is more than what is required of the legitimate parties, yet less than what was required of the classical eavesdropper.

The introduction of quantum computers seems to be for the advantage of the eavesdroppers. Can we remedy this situation? Is any security possible at all against a quantum eavesdropper if both legitimate parties are restricted to use classical computers? The next chapters, in which we study these and other related questions, will reveal *counter-intuitive* answers.

CHAPTER 4

QUANTUM PROTOCOLS AGAINST QUANTUM ADVERSARIES

Merkle's scheme, which provides provably quadratic security against classical eavesdroppers [8], has no security at all in a quantum world. Our notion of "security" is defined in Section 2.7. Allowing Bob to be quantum, Brassard and Salvail [23] provided a scheme in which a quantum eavesdropper needs $\Omega(\sqrt{N^3}\,) = \Omega(N^{3/2})$ quantum queries [19, 20]. Subsequently, it remained open the following question: *Is $\Omega(N^{3/2})$ the optimal key agreement security in the random oracle model, when legitimate parties make use of quantum computations?*

Part of our contributions, this chapter consists of *two* novel provably secure protocols that answer positively this question. Both protocols provide security of $\Omega(N^{5/3})$, though they are based on two different but *equivalent* search problems.

Before presenting any of the protocols, we remind of the setting in this chapter. Legitimate parties, Alice and Bob, are allowed to make quantum computations. In the first protocol both parties are quantum while in the second one only Bob needs to be quantum. In both cases, the eavesdropper is assumed to have unrestricted quantum resources in addition to having full knowledge of any communicated message on the unique *classical* authenticated channel. All parties have access to the same random oracle and our measure of complexity is the *query complexity.* Note finally that all our results are implicitly stated "up to logarithmic factors".

4.1 The First $\Theta(N^{5/3})$ Quantum Protocol

We describe this novel protocol assuming the existence of *two* black-box functions $f : [N^3] \rightarrow [N^k]$ and $g : [N^3] \times [N^3] \rightarrow [N^{k'}]$ that can be accessed in superposition of inputs.

The constants k and k' are chosen large enough so that neither f nor g has a collision in their images (both functions are one-to-one), except with polynomially vanishing probability, or equivalently, with probability in $o(1)$. This assumption is satisfied whenever the range size is quadratically larger than the domain size, taking into account the birthday problem (Section 2.8.6). Specifically, we take $k > 6$ and $k' > 12$, based on our calculations in Section 5.3.4. For simplicity, we shall systematically disregard the possibility that such collisions might exist.

The reason for which we left the notion *negligible*, which is necessary in standard cryptography, in favour of *vanishing* is explained in Section 5.3.1. In Section 2.2, we explain the meaning of the notions *random function* or *how to choose* a function at random. In Section 2.3, we define the (random) *oracles*.

The notion of *black-box functions*, which is fundamental to understand in this work, is defined and explained in Section 2.4. For instance, the black-box function f is selected at random from the set of all mappings from $D = N^3$ into $R = N^k$. It might be useful to think of it as a tuple $y = (y_1, \cdots, y_{N^3})$ of integers where each integer was chosen independently and uniformly at random, that is, $y \in [N^k]^{N^3}$. In addition, for all $1 \leqslant i \leqslant N^3$, this box outputs $y_i \in N^k$ on an input $i \in [N^3]$.

Note that *a single binary random oracle* (which "implements" a random function from the integers to $\{0, 1\}$) could be used to define both functions f and g provided we disregard logarithmic factors in our analyses since $O(\log N)$ queries to the random oracle would suffice to compute f or g on any single input. Indeed, to specify function f for instance one needs $N^3 \log N^k$ bits. Think of the N^3 points as embedded in the oracle in a canonical form. Each image is represented by $\log N^k$ bits, and each query $i \in [N^3]$ for f requires $\log N^k$ queries to its corresponding binary oracle to construct the integer $f(i) \in [N^k]$. It is understood hereinafter that all our results are implicitly stated "up to logarithmic factors".

On the other hand, multiple oracles can be represented using a single oracle by pre-pending a fixed bit-string to the beginning of each query. For instance queries of the form "$0i$" and "$1i$" can be considered queries to two separate functions f and g respectively.

Protocol 1 (Quantum parties vs quantum adversaries)**.**

1. *Alice picks at random N distinct points, $x_1, x_2, \ldots, x_N$, in the domain of f and transmits them encrypted, by querying the black-box function f, to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$ be the set of random points, which is kept secret by Alice, and let $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$ be the set of encrypted points sent from Alice to Bob. Keep in mind that Alice knows both X and Y whereas Bob and the eavesdropper have knowledge of Y only without querying the oracle f.*

2. *Bob finds the pre-images x and x' of* two *distinct random elements in Y, according to the following procedure. He sorts the elements of Y so that, given any y, he can efficiently determine whether or not $y \in Y$. Then, he defines the Boolean function $\phi : [N^3] \to \{0, 1\}$ as follows:*

$$\phi(x) = \begin{cases} 1 & \text{if } f(x) \in Y \\ 0 & \text{otherwise}. \end{cases}$$

   The problem is then reduced to finding a pre-image x for which $\phi(x) = 1$. There are exactly N solutions satisfying this condition, out of N^3 points in the domain of ϕ. Using generalized Grover's algorithm (BBHT), Bob can find a solution in $O(\sqrt{N^3/N}\,) = O(N)$ quantum queries to function ϕ, or equivalently to f, since each query to ϕ implies one and only one query to f. He needs to repeat this procedure twice in order to get both x and x'. (A small variation in function ϕ can be used the second time to make sure that $x' \neq x$).

3. *Bob sends back $w = g(x, x')$ to Alice such that $x < x'$.*

4. *Since Alice had kept her random secret set X, there are only N^2 candidate pairs $(x_i, x_j) \in X \times X$ such that $g(x_i, x_j)$ could equal w. Using Grover's algorithm, she can find Bob's pair (x, x') with $O(\sqrt{N^2}\,) = O(N)$ queries to function g. The secret established by Alice and Bob is the pair (x, x').*

We first verify that the protocol is valid, in the sense that at the end of its execution, Alice and Bob agree on a secret after a number of queries linear in N.

Indeed, Alice makes N classical queries to f in Step 1 and $O(N)$ quantum queries to g in Step 4, whereas Bob makes $O(N)$ quantum queries to f in Step 2 and a single classical query to g in Step 3. If the protocol is constructed over a binary random oracle, it will have to be called $O(N \log N)$ times since it takes $O(\log N)$ binary queries to compute either function on any given input.

It is crucial to know that Bob, to find $x \in X$, does *not* choose $y \in Y$ randomly and inverts it afterwards, which would require $\Omega(\sqrt{N^3}\,)$ and consequently make the protocol invalid. Instead, he applies BBHT generalisation an appropriate number of times in order to evolve the initial state, *superposition of all possible inputs*, to a final state, very close to a *superposition of all possible solutions*. Then, he measures the final state, producing a purely random image, according to the axioms of quantum mechanics, coupled with its corresponding pre-image.

4.1.1 Quantum Attack

All the cryptanalytic attacks against this scheme, such as direct application of Grover's algorithm, generalized Grover's algorithm, or even more sophisticated attacks based on amplitude amplification [26], require of the eavesdropper $\Omega(N^2)$ quantum queries to functions f and/or g. These quantum search algorithms are reviewed in Section 2.12.

However, a more powerful attack based on the recent paradigm of quantum walks [5, 36, 59, 70, 75] allows the eavesdropper to learn Alice and Bob's key (x, x') with an expected $O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N)$ queries to g. Actually, our attack combines Ambainis' algorithm for element distinctness [4] with Magniez and Nayak's quantum walk algorithm for testing group commutativity [58].

To analyse the cost of our quantum algorithms, we will always apply the theorem of Magniez, Nayak, Roland and Santha (MNRS) [59]. Actually, the same upper bounds can also be obtained using Szegedy's theorem [75]. Both theorems are reviewed in Section 2.12.5.

Theorem 4.1.1. *There exists an eavesdropping strategy that outputs the pair (x, x') in Protocol 1 with $O(N^{5/3}\sqrt{\log N})$ expected quantum queries to functions f and g.*

Proof. The attack was originally inspired by Ambainis' quantum algorithm for element distinctness [4], which can find the (single) pair (i, j) such that $\xi(i) = \xi(j)$ with $O(N^{2/3})$ expected queries to a function ξ whose domain consists of N elements. In a nutshell, we apply Ambainis' algorithm for element distinctness with two modifications: (1) instead of looking for i and j such that $\xi(i) = \xi(j)$, we are looking for x and x' such that $g(x, x') = w$; and (2) instead of being able to get randomly chosen values in the image of ξ with a single query to oracle ξ per value, we need to get random elements of X by applying BBHT on the list Y, which requires $O(\sqrt{N^3/N}) = O(N)$ queries to oracle f per element. From the first modification, it might be clear that this problem is a special case of the subset finding problem [36] (see Section 2.8.3). The second modification explains why the number of queries to f, compared to $O(N^{2/3})$ queries to ξ for element distinctness, will be multiplied by $O(N)$. To determine the query complexity of our quantum attack, we need to introduce few ingredients.

It turns out that the special structure of our problem (composition of functions) prevents from implementing a quantum walk on Johnson graphs, which is used in element distinctness algorithm or its generalization [4, 36]. Therefore, our eavesdropping algorithm is combined with a quantum walk algorithm for testing group commutativity due to Magniez and Nayak [58].

We briefly introduce the random as well as the quantum walk on this graph. Let X denote the set of elements whose images are sent by Alice. The random walk takes place over a graph $G(V, E)$ whose vertices are the r-tuples of distinct elements of X and eigenvalue gap is $\delta \in \Omega(1/r \log r)$—see Section 2.12.6 for a review of the random walk on this graph. More precisely, each vertex u has the form $|u_1 u_2 \ldots u_r\rangle$ where $u_i \in X$ and $u_i \neq u_j$ for all $1 \leqslant i \neq j \leqslant r$. The value of the parameter r will be determined during the analysis of the algorithm. Two vertices, u and v, are connected if and only if v is obtained from u by permuting two of its components or replacing one of them by an element of X not in the node.

The algorithm initially starts with a superposition state of all vertices $u \in V$, which can be done by preparing each u_i in superposition. We are looking for a vertex that contains the two elements x and x' of X such that $g(x, x') = w$, where w is the value announced by Bob in Step 3 of the protocol.

To define the quantum walk on this graph, let $\mathcal{H}_V$ be the Hilbert space whose orthonormal basis states are the elements of V. We define a basis state $|u\rangle$ for each $u \in V$. Since the walk takes place on the edges instead of vertices, the full quantum state of the algorithm has the form $|u\rangle \otimes |v\rangle$ where $u, v \in V$. One step of the quantum walk (which is actually two steps on the graph) is a product of two unitary transformations on the space $\mathcal{H}_V \otimes \mathcal{H}_V$, which is in fact a subspace of the algorithm's space (as it might be clear in the setup phase later).

Thanks to the theorem of Szegedy [75] or MNRS [59], the random walk P on this graph can be quantized. We need to know the eigenvalue gap $\delta = \delta(P)$ of the random walk, and the fraction of marked vertices under the stationary distribution denoted by ε. Afterwards, the cost of the resulting quantum algorithm can be written as a function of the quantum costs S, U and C:

Setup cost S: corresponds to prepare the state

$$\sum_{u \in V} \sqrt{\pi_u}|u\rangle \otimes |0\rangle$$

where π is the uniform distribution of P, and the all-zero state $|0\rangle$ belongs to V. Equivalently, the cost S corresponds to finding r random distinct elements of X. To find *one* such element, we apply BBHT's algorithm, which takes $O(N)$ queries to f even to find an element of X guaranteed to be different from those already in the initial vertex (provided $k \ll N$, which it will be). Therefore, $\mathsf{S} = O(rN)$ quantum queries to f. Here are more details.

1. Prepare the following state:

$$|\psi_1\rangle = \left(\frac{1}{\sqrt{N^3}} \sum_{u_1 \in [N^3]} |u_1\rangle\right) \otimes \left(\frac{1}{\sqrt{N^3}} \sum_{u_2 \in [N^3]} |u_2\rangle\right) \otimes \cdots \otimes \left(\frac{1}{\sqrt{N^3}} \sum_{u_r \in [N^3]} |u_r\rangle\right).$$

It consists of r independent registers. Each register is prepared initially in superposition state of all possible integers in $[N^3]$ without involving oracle queries, which can be done by applying Hadamard gates on the *all-zero* state.

2. Produce the superposition state of the $r!\binom{N}{r}$ possible vertices of the graph:

$$|\psi_2\rangle = \frac{1}{\sqrt{N}} \sum_{u_1 \in X} |u_1\rangle \otimes \left(\frac{1}{\sqrt{N-1}} \sum_{u_2 \in X}^{u_2 \neq u_1} |u_2\rangle \cdots \otimes \left(\frac{1}{\sqrt{N-r}} \sum_{u_r \in X}^{u_r \neq u_i; 1 \leqslant i < r} |u_r\rangle \right) \right).$$

To construct $|\psi_2\rangle$, we proceed as follows. First, we produce a uniform superposition of all elements $u_1 \in X$ by applying BBHT on the first register of $|\psi_1\rangle$. Since we know the exact number of solutions, which is N at this step, this state can be produced with certainty in $O(\sqrt{N^3/N} = N)$ queries even in the worst case, thanks to Theorem 2.12.1. Second, for each u_1 being in superposition state, we produce a superposition state of all $u_2 \in X$ such that $u_2 \neq u_1$, by applying BBHT on the second register with $N-1$ as the number of solutions this time. We continue similarly until the rth step in which, for each $(r-1)$-tuple $|u_1 u_2 \ldots u_{r-1}\rangle$, we produce a uniform superposition state of elements $u_r \in X$ such that $u_r \neq u_i$ for $1 \leqslant i < r$. This step can be achieved in $O(\sqrt{N^3/N-r}) = O(N)$ queries, since the number of solutions is now $N - r \approx N$.

Update cost U: corresponds to realize any of the following unitary transformations and their inverses:

$$\begin{aligned} U_1 : |u\rangle \otimes |0\rangle &\rightarrow |u\rangle \otimes \left(\sum_{v \in V} \sqrt{p_{uv}} |v\rangle \right), \\ U_2 : |0\rangle \otimes |v\rangle &\rightarrow \left(\sum_{u \in V} \sqrt{p_{vu}} |u\rangle \right) \otimes |v\rangle. \end{aligned}$$

The graph is symmetric, meaning that $p_{uv} = p_{vu}$ for all $u, v \in V$. Besides, the distribution of the random walk is uniform: the probability of moving from $|u, v\rangle$ to $|u, v'\rangle$ is the same, for all $u \in V$ and for all neighbouring vertices $v' \neq v$.

Besides, the result of BBHT's algorithm is a state of the N possible elements of X with the same amplitude (uniform superposition), which is exactly what is needed to realize U_1 and U_2 (see definition of the random walk in Section 2.12.6). Therefore, each application of either U_1 or U_2 invokes BBHT only once, thus taking $O(N)$ queries.

Checking cost C: corresponds (in this protocol) to the phase flip operation, which is necessary to distinguish the marked r-tuple, that is, the unique vertex containing the pair (x, x') such that $g(x, x') = w$. This unitary is defined as follows:

$$F|u\rangle = \begin{cases} -|u\rangle & \text{if } x, x' \in u, \\ |u\rangle & \text{otherwise.} \end{cases}$$

This phase can be achieved using Grover's algorithm. Since there are r^2 possible pairs of elements in the vertex, it can be done in $O(\sqrt{r^2}\,) = O(r)$ quantum queries to g.

To analyse the cost of our quantum algorithm on this graph, we can apply either Szegedy's or MNRS' theorem, since the chain is symmetric and reversible in addition to be ergodic—see Section 2.12.5 for a review of this topic.

Be aware that our cryptanalysis algorithm runs over only one copy of the (basic) quantum. Maniez and Nayak [58] used two independent simultaneous copies in their original algorithm. However, the graph maintains the same mathematical properties, specially $\delta \in \Omega(1/r \log r)$ and $\varepsilon \in \Omega(r^2/N^2)$.

It is worth mentioning how to calculate ε, which is the probability that a random vertex is marked (contains a solution). A vertex is marked in this case if it contains elements x, x' such that $g(x, x') = w$. Assume that such a pair exists and consider a random r-tuple vertex $u \in V$. Then, the probability of u being marked is

$$\begin{aligned} \Pr[x, x' \in u] &= \Pr[x \in u] \cdot \Pr[x' \in u | x \in u] \\ &= \frac{r}{N} \cdot \frac{r-1}{N-1} \\ &\approx \frac{r^2}{N^2} \end{aligned}$$

Note that this result can also be obtained using the usual combinatorial method.

Putting all the necessary ingredients together and using the MNRS theorem, the expected cryptanalytic cost is in the order of:

$$\begin{aligned}
& \mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right) \\
= \;& \mathsf{S} + \tfrac{N}{r}\left(\sqrt{r\log r}\,\mathsf{U} + \mathsf{C}\right) \\
= \;& \Big(rN \text{ queries to } f\Big) + \tfrac{N}{r}\Big(\sqrt{r\log r}(N \text{ queries to } f) + (r \text{ queries to } g)\Big) \\
= \;& \left(rN + \frac{N^2}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } f \;\; \textbf{and} \;\; N \text{ queries to } g\,.
\end{aligned}$$

To minimize the number of queries to f, we choose r so that $rN = N^2/\sqrt{r}$, which is $r = N^{2/3}$. It follows that a quantum eavesdropper can find the key (x, x') with an expected $O(\frac{N^2}{\sqrt{r}}\sqrt{\log r}\,) = O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N)$ queries to g.

Note that the use of Grover's algorithm in the checking phase was not necessary to prove Theorem 4.1.1. Should this step be carried out classically, this would result in $\mathsf{C} = O(r^2)$ queries to g. Consequently, the expected cost to find the key would become $O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N \cdot r) = O(N^{5/3})$ queries to g.

However, if we apply Szegedy's theorem, then the cost becomes in the order of:

$$\begin{aligned}
& \mathsf{S} + \frac{1}{\sqrt{\delta\varepsilon}}\Big(\mathsf{U} + \mathsf{C}\Big) \\
= \;& \Big(rN \text{ queries to } f\Big) + \frac{N}{r}\sqrt{r\log r}\Big(N \text{ queries to } f + r \text{ call to } g\Big) \\
= \;& \left(rN + \frac{N^2}{\sqrt{r}}\sqrt{\log r}\right) \text{queries to } f \;\; \textbf{and} \;\; \Big(N\sqrt{r\log r}\Big) \text{ queries to } g\,.
\end{aligned}$$

To minimize the number of queries to f and g, we choose $r = N^{2/3}$. It follows that the total cost is $O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N^{4/3}\sqrt{\log N})$ queries to g. Clearly, this algorithm requires more queries to function g. Note also that Szegedy's algorithm can determine rather than find a solution. □

4.1.2 Lower Bound

We prove in this section that the preceding quantum attack against our quantum protocol is optimal, (up to the square-root of a logarithmic factor), by this theorem.

Theorem 4.1.2 (Eavesdropping lower bound). *Any eavesdropping strategy $\mathcal{A}$ that learns the key (x, x') in Protocol 1 requires a total of $\Omega(N^{5/3})$ quantum queries to functions f and g. Any strategy $\mathcal{A}$ asking $o(N^{5/3})$ queries can find the key* only *with $o(1)$ probability over the random views of the protocol.*

The proof of Theorem 4.1.2 consists of four steps:

1. We define a composed search problem H related to the hardness of breaking our protocol;
2. We prove a lower bound on the difficulty to solve H (Lemma 1), using a new composition theorem for the generalized adversary method [29], which is reviewed in Section 2.13.1;
3. We reduce H to a less structured problem H' (Lemma 2), giving the same desired lower bound; and
4. We reduce the problem H' to the eavesdropping problem against our protocol. More precisely, we show that any attack on our key agreement scheme that would have a non-vanishing probability of success after $o(N^{5/3})$ queries to functions f and/or g could be turned into an algorithm capable of solving H' more efficiently than possible.

In Step 1, we compose the element distinctness problem (ED) with N instances of a search problem with a promise ($\mathsf{pSEARCH}$), which results in the starting search problem $\mathsf{H} = \mathsf{ED} \circ \mathsf{pSEARCH}^N$. We first recall of these problems or their variants.

Consider an oracle function $\xi : [N] \to [M]$ such that there exists a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $\xi(i) = \xi(j)$. Ambainis' quantum algorithm for element distinctness [4] or its generalization [36] (Section 2.12.4) can find this pair with $O(N^{2/3})$ queries to function ξ and Aaronson and Shi proved that this is optimal even for the decision version of this problem [2].

Consider the promise problem $\mathsf{pSEARCH} : P \to [M]$ with $P \subseteq ([M]')^{N^2}$. Recall that $[M]'$ denotes $\{0\} \cup [M]$. On input $(a_1, \ldots, a_{N^2})$, the promise P is that all but one of the values are zero. The goal is to find and output this nonzero value by making queries that return a_i on input i.

The structure of problem H allows us to think of it as a two-level tree: the root being labelled by ED (or SF) and each of the κ leaves being labeled by $\mathsf{pSEARCH}$. An input $x \in P^N$ to H can be thought of as being N parts, $x = (x_1, \ldots, x_N)$. We evaluate H on x by first finding the N non-zero values, and then computing ED on those values. More formally, on input $x \in P^N$,

$$\mathsf{H}(x) = \mathsf{ED}(\mathsf{pSEARCH}(x_1), \ldots, \mathsf{pSEARCH}(x_N)).$$

Consider now a function $h : [N] \times [N^2] \to [M]'$. The domain of this function is composed of N "buckets" of size N^2, where $h(i, \cdot)$ corresponds to the i^{th} bucket for $1 \leqslant i \leqslant N$. In bucket i, all values of the function are 0 except for *one single* random $x_i \in [N^2]$ for which $h(i, x_i) = \xi(i)$. Symbolically,

$$h(i,j) = \begin{cases} \xi(i) & \text{if } j = x_i \\ 0 & \text{otherwise}\,. \end{cases}$$

It follows from the definitions of ξ and h that there is a single pair of distinct elements a and b in the domain of h such that $h(a) = h(b) \neq 0$. How difficult is it to find this pair given an oracle access for function h but *no* direct access to ξ? We answer this question by the following lemma, achieving the second step of the proof.

Lemma 1 (Lower bound for h)**.** *Given h structured as above, finding the pair of distinct elements a and b in the domain of h such that $h(a) = h(b) \neq 0$ requires $\Omega(N^{5/3})$ quantum queries to h. Any algorithm $\mathcal{A}$ making $o(N^{5/3})$ queries solves this problem* only *with $o(1)$ probability over the randomness of considered oracles.*

Proof. The search problem can be modelled as the composition of element distinctness across buckets, the *outer* function, with finding the single non-zero entry in each bucket, the *inner* function. In other words, it is a problem of *searching* among N^2 possibilities for the unique non-zero $h(i, \cdot)$ for each i, and then *finding two* of those N elements that are *equal.*

One would like to apply a *composition theorem* for the generalized adversary method, which would give that the quantum query complexity of h is the product of the quantum query complexities of the outer function and the inner function, or symbolically, $Q_2(h) = \Omega(N^{2/3}\sqrt{N^2}\,)$. Høyer, Lee, and Špalek [81] proved the first composition theorem, which requires not only the inner function to be Boolean but also the outer one. This limitation was partially removed by Lee, Mittal, Reichard, Špalek, and Szegedy [56] whose theorem requires only the inner function to be Boolean. Unfortunately, both composition theorems require the *inner* function to be Boolean, which is not the case here for pSEARCH. Trying to make the inner function Boolean violates some conditions of these theorems. Therefore, we use a new composition theorem [29] based on similar techniques (see Section 2.13.1). In particular, the problem becomes a special case of technical Theorem 2.13.8 with parameters $\kappa = N$ (the number of buckets) and $\eta = N^2$ (the size of the buckets). Using Theorem 2.13.6 along with the quantum query complexities for element distinctness and pSEARCH, it follows that finding the desired pair (a, b) requires

$$\Omega(\kappa^{2/3}\eta^{1/2}) = \Omega(N^{2/3}\sqrt{N^2}\,) = \Omega(N^{5/3})$$

quantum queries to h. □

Be aware that, in order to apply the composition theorem, it is necessary to know the adversary bounds of problems under consideration. However, this is not the case for element distinctness whose lower bound was found by the polynomial method [10]. A recent theorem of Ref. [56] shows that the generalized adversary bound is tight for total and partial functions. Therefore, we may conclude that there exists an $\Omega(\kappa^{2/3})$ adversary bound for element distinctness.

For Step 3, consider a slightly less structured search problem in which there are no longer buckets, but there is an added coordinate in the image of the function:

$$h' : [N^3] \to [N]' \times [M]'$$

We justify the added coordinate $[N]'$ in the last step of the proof where it turns out to be necessary. Define this function such that $h'(a) = (0, 0)$ on all but N randomly chosen points denoted by w_1, w_2,..., w_N. On these N points, $h'(w_i) = (i, \xi(i))$, where ξ is the function for element distinctness considered at the beginning of Step 1. We are required to find the unique pair of distinct a and b in $[N^3]$ such that $\pi_2(h'(a)) = \pi_2(h'(b)) \neq 0$, where "$\pi_2$" denotes the projection on the second coordinate. Similarly, "π_1" denotes the projection on the first coordinate.

The lower bound on the earlier search problem concerning h implies directly the same lower bound on the new search problem concerning h' since any algorithm capable of solving the new problem can be used at the same cost to solve the earlier problem through randomization and some technical adjustment. In other words, the more structured version of the problem cannot be harder than the less structured one. The next lemma formalizes this argument.

Lemma 2 (Lower bound for h')**.** *Given h' structured as above, finding the pair of distinct preimages a and b of h' such that $\pi_2(h'(a)) = \pi_2(h'(b)) \neq 0$ requires $\Omega(N^{5/3})$ quantum queries to h'.*

Proof. Define intermediary function $\tilde{h} : [N] \times [N^2] \to [M]' \times [M]'$ by

$$\tilde{h}(i,j) = \begin{cases} (i, h(i,j)) & = & (i, \xi(i)) & \text{if } h(i,j) \neq 0 \\ (0, h(i,j)) & = & (0,0) & \text{otherwise}\,. \end{cases}$$

It is elementary to reduce the search problem concerning h to the one concerning $\tilde{h}$ as well as the search problem concerning $\tilde{h}$ to the one concerning h'. (However, we describe it briefly for completeness when presenting Protocol 2.) Therefore, the lower bound concerning h given by Lemma 1 applies *mutatis mutandis* to h'. □

Finally, it remains to finish the last step of Theorem 4.1.2, which is to reduce the search problem concerning h' to the cryptanalytic difficulty against our protocol.

Proof of Theorem 4.1.2. Consider any eavesdropping strategy $\mathcal{A}$ that listens to the communication between Alice and Bob and tries to determine the key (x, x') by querying black-box functions f and g. In fact, there are no Alice and Bob here. Instead, there is an oracle function $h' : [N^3] \to [N]' \times [M]'$ as described before, for which we want to solve the search problem by using unsuspecting $\mathcal{A}$ as a resource. In other words, given access to h' only, we have to simulate for $\mathcal{A}$ an environment identical to a random view of the protocol, except with vanishing probability. The main idea is to supply $\mathcal{A}$ with a completely fake "conversation" between "Alice" and "Bob" as follows.

For sufficiently large k and k', we choose at random N points y_1, y_2,…, y_N in $[N^k]$ and one point $w \in [N^{k'}]$, and we pretend that Alice has sent the y_i's to Bob and that Bob has responded with w. Let $\hat{Y}$ denote the subset containing the points y_1, y_2,…, y_N. In addition, we choose random functions $\hat{f} : [N^3] \to [N^k]$ and $\hat{g} : [N^3] \times [N^3] \to [N^{k'}]$. Note that the selection of $\hat{f}$ and $\hat{g}$ may take a lot of *time*, but this does not count towards the number of queries that will be made to function h', and our lower bound on the search problem concerns only this number of queries. We may prefer to choose randomly values of $\hat{f}$ and $\hat{g}$ one by one (dynamic viewpoint that is usually used in the case of classical cryptography), which is not possible for the following reason. In quantum algorithms, queries are usually asked in superposition of all possible inputs; in order to create interference. Therefore, the function specification must be available before starting the reduction. The final thing that we need to make the reduction is a random Boolean $s \in \{\mathsf{true}, \mathsf{false}\}$. The Boolean s indicates, when true (resp. false), that the fake "execution" is such that "Bob" has first picked x and then x' such that $x < x'$ (resp. $x' > x$). Both cases happen with probability $^1/_2$ in any real execution and for any public announcements Y and w. The value s will be used in the reduction to distinguish between $g(x, x')$ and $g(x', x)$ so that only $g(x, x')$ will be set to w.

The reduction $\mathcal{A}'$, which uses $\mathcal{A}$ as a subroutine, is derived as follows

We wait for $\mathcal{A}$'s queries to f and g.

- When $\mathcal{A}$ asks for $f(i)$ for some $i \in [N^3]$, there are two possibilities:
  - If $h'(i) = (0, 0)$, return $\hat{f}(i)$ to $\mathcal{A}$ as value for $f(i)$.
  - Otherwise, return $y_{\pi_1(h'(i))}$.
- When $\mathcal{A}$ asks for $g(i, j)$ for some $i, j \in [N^3]$, there are again two possibilities:
  - If $\pi_2(h'(i)) = \pi_2(h'(j)) \neq 0$ and either s is true and $i < j$ or s is false and $i > j$, return w as value for $g(i, j)$.
  - Otherwise, return $\hat{g}(i, j)$.

At this stage, it is convenient to explain the utility of the left-hand component in the image of h'. For each $h'(i) \neq (0, 0)$, the algorithm $\mathcal{A}$ should get one of the points of $\hat{Y}$; generated at the beginning of this "artificial" cryptanalytic task. There is a one-to-one correspondence between the y_i's and the non-zero values in the image of h. Without the added coordinate, we would use the value $h'(i) = \xi(i)$ as index to return the corresponding point $y_{\xi(i)}$, and concequently, $\mathcal{A}$ would get at some time the same point $y_{\xi(i)}$ for two distinct elements i, j whenever $\xi(i) = \xi(j)$. The reduction becomes inconsistent with the real-world execution of the protocol. In contrast, adding this component, which takes values in $[N]'$, guarantees to have N indices form 1 to N in addition to the 0 value, which enables us to access any y_i consistently with the real world. This solves this problem as shown in the reduction. If $\mathcal{A}$ was classical, one would simply solve this problem using a table that keeps track of any $h'(i) \neq (0, 0)$. However, maintaining such a process in the quantum case seems to be difficult or probably impossible.

When selecting $\hat{Y} = \{y_1, \ldots, y_N\}$ from $[N^k]$ and $\hat{f} : [N^3] \to [N^k]$, it may happen that there is a specific y_i that is also an image of $\hat{f}$ on some input, making the reduction inconsistent with the real world execution. However, this event happens with probability smaller than $N \cdot N^3/N^k = N^4/N^k$, which vanishes super-quadratically since $k > 6$. Provided that there is neither collision in $\hat{Y}$ nor in $\hat{f}$, this bound can be easily obtained when we think of this event as a birthday problem between two subsets: $\hat{Y}$ and N^3 random points in N^k (see Section 5.3.4).

Suppose now that $\mathcal{A}$ correctly returns the pair (i, j) for which it was told that $g(i, j) = w$, which is what a successful eavesdropper is supposed to accomplish. This pair is in fact the answer to the search problem concerning h' since $g(i, j) = w$ implies that $\pi_2(h'(i)) = \pi_2(h'(j)) \neq 0$, except with the vanishing probability that $\hat{g}(i', j') = w$ for some query (i', j') that $\mathcal{A}$ asks about g. This event happens with vanishing probability $N^6/N^{k'}$ since $k' > 12$, considering the birthday problem between two subsets having cardinalities 1 and N^6. In fact, we always assume that the function range size is at least quadratically larger than the domain size.

Queries asked by $\mathcal{A}$ concerning f and g are answered in the same way as they would be if f and g were two random functions consistent with the Y and w announced by Alice and Bob during the execution of a real protocol. Indeed, remember that Y (subset of $[N^k]$) and w (element of $[N^{k'}]$) are uniformly picked at random in both the simulated and the real worlds. Moreover, the simulated function f is such that $f(i)$ is random when $h'(i) = (0, 0)$. The remaining N output values are in Y, as expected by $\mathcal{A}$. On the other hand, the simulated function g is random everywhere except for one single input pair (i, j), $i \neq j$ for which $g(i, j) = w$, as it is also expected by $\mathcal{A}$. Therefore, $\mathcal{A}$ will behave in the environment provided by the simulation exactly as in the real world. Since we disregard the vanishing possibility that g might not be one-to-one, the reduction solves the search problem concerning h' whenever $\mathcal{A}$ succeeds in finding the key. Notice that each (new) question asked by $\mathcal{A}$ to either f or g translates to one or two questions actually asked to h'. This mainly happens when querying $g(i, j)$ for some positive integers $i \neq j$, which requires querying $h(i)$ and $h(j)$.

It follows that any successful cryptanalytic strategy that makes $o(N^{5/3})$ total queries to f and g would solve the search problem with only $o(N^{5/3})$ queries to function h', which is impossible, except with vanishing probability. This establishes the $\Omega(N^{5/3})$ lower bound on the cryptanalytic difficulty of breaking our protocol, again except with vanishing probability over the random views of the protocol, matching the upper bound (up to a logarithmic factor) provided in Section 4.1.1. □

4.2 The Second $\Theta(N^{5/3})$ Quantum Protocol

As the title indicates, the second quantum protocol is as secure as Protocol 1. However, it has several new features that stem from using the $\oplus$ operation instead of a random oracle at the third step, where "$\oplus$" is the bitwise exclusive-or. We mention here briefly these features but their interest will be clarified as we advance.

Using $\oplus$ will allow us to design more secure protocols than those based on element distinctness (or subset-finding). More precisely, this will enable us to generalize the classical as well as the quantum protocols, providing further more improvements. Second, the security proof of those protocols will be *exactly* the same as the one in this section. Third, in the quantum setting, Alice can remain classical while preserving the same security level. It also results in the complexity relationship between element distinctness and 2XOR, which is important in its own. We now proceed with the second scheme.

Similarly to Protocol 1, we assume the existence of *two* random oracle functions $f : [N^3] \to [N^k]$ and $t : [N^3] \to [N^{k'}]$ that can be accessed in quantum superposition of inputs.

The constant k is chosen large enough so that f is one-to-one, except with polynomially vanishing probability. But, the condition on k' is slightly different. It is chosen so large that t is one-to-one, and that $t(a) \oplus t(b) \oplus t(c) \oplus t(d) \neq 0$ whenever $\{a, b, c, d\}$ contains at least three distinct elements in the domain of t, except with vanishing probability. In other words, for any integer w, the event $t(a) \oplus t(b) = w$ and $t(c) \oplus t(d) = w$ should only happen with vanishing probability. Finding such a collision is equivalent to finding one in a function $h : [N^6] \to [N^{k'}]$. Hence, the problem becomes a special case of Theorem 6.1.1. The probability of this event is at most $N^{12}/N^{k'}$, which vanishes for any $k' > 12$. For simplicity, we shall systematically disregard the possibility that such exceptions might occur. Without delay, we describe our second protocol; the first two steps remain exactly the same as in Protocol 1.

Protocol 2 (Quantum parties vs quantum adversaries)**.**

1. *Alice picks at random N distinct points, $x_1, x_2, \ldots, x_N$, in the domain of f and transmits them encrypted, by querying the black-box function f, to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$ and $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$.*
2. *Bob finds the pre-images x and x' of* two *distinct random elements in Y. To find each one of them, he uses generalized Grover's algorithm, which finds a solution in $O(\sqrt{N^3/N}\,) = O(N)$ quantum queries to f.*
3. *Bob sends back $w = t(x) \oplus t(x')$ to Alice.*
4. *Alice queries the oracle t on her randomly chosen set X that she has kept. There are only N^2 candidate pairs $(x_i, x_j) \in X \times X$ such that $t(x_i) \oplus t(x_j)$ could equal w. Therefore, she can find the unique pair (x_i, x_j) without any additional queries. Alice and Bob's secret is the pair (x, x'), assuming $x < x'$.*

All counted, Alice makes exactly N classical queries to f in Step 1 and N classical queries to t in Step 4, whereas Bob makes $O(N)$ quantum queries to f in Step 2 and two classical queries to t in Step 3. Therefore, legitimate parties make a total of $O(N)$ quantum and/or classical queries, thus the protocol is valid.

Note that our measure of complexity is the query complexity. However, if we also care about time complexity, it seems at first that Alice needs to try about half the N^2 pairs. But, this can easily be done in linear time (see Section 6.4).

4.2.1 Quantum Attack

The same previous attack enables Eave to recover Alice and Bob's key (x, x') with an expected $O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N^{2/3}\sqrt{\log N})$ queries to t, although the property for which we are looking is different. This is the second manifestation showing that Ambainis' algorithm can indeed be used beyond element distinctness.

Theorem 4.2.1. *There exists an eavesdropping strategy that outputs the pair (x, x') in Protocol 2 with $O(N^{5/3}\sqrt{\log N})$ expected quantum queries to functions f and t.*

Proof. An eavesdropper can set up a quantum walk similar to that in Section 4.1.1, except that now (1) instead of looking for x and x' such that $g(x, x') = w$, we are looking for x and x' such that $t(x) \oplus t(x') = w$ where w is the value announced by Bob in Step 3 of the protocol; and (2) instead of being able to get randomly chosen values in the image of ξ with a single query per value, we need to get random elements of X by applying BBHT on the list Y and queries the oracle t on them; and (3) the algorithm maintains a data structure to store the values under t along with the r-tuples.

We apply Theorem 2.12.4 to analyse the cost of quantum walk algorithm on this graph. The set up cost S corresponds to finding r random elements of X using BBHT, and querying t on them. Therefore, the setup cost is $\mathsf{S} = O(rN)$ queries to f and r queries to t. The update cost corresponds to finding one random element of X not already in the node and querying t on it, which is $\mathsf{U} = O(N)$ queries to f and *one* query to t. The checking cost C requires us to decide if there is a pair (x, x') of elements in the node such that $t(x) \oplus t(x') = w$, which is done without any additional queries because the values under t are already in the vertex (data structure) and $\oplus$ is used instead of an oracle. Therefore, the checking cost $\mathsf{C} = 0$, contrasting with that of Section 4.1.1, which requires $O(r)$ queries.

Putting all necessary ingredients together and applying the MNRS's theorem, the expected cryptanalytic cost is in the order of:

$$\begin{aligned} & \mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right) \\ = \;& \mathsf{S} + \tfrac{N}{r}\left(\sqrt{r\log r}\,\mathsf{U}\right) \\ = \;& \left(rN + \frac{N^2}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } f \;\; \textbf{and} \;\; \left(r + \frac{N}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } t\,. \end{aligned}$$

To optimize the number of queries to f and t, we choose r so that $rN = N^2/\sqrt{r}$, implying $r = N^{2/3}$. It follows that a quantum eavesdropper can find the key (x, x') with an expected $O(N^{5/3}\sqrt{\log N})$ queries to f and $O(N^{2/3}\sqrt{\log N})$ queries to t. □

4.2.2 Lower Bound

The impossibility of finding the key (x, x') with fewer than $\Omega(N^{5/3})$ queries to f and/or g, except with vanishing probability, is formalized by the following theorem.

Theorem 4.2.2 (Eavesdropping lower bound)**.** *Any eavesdropping strategy $\mathcal{A}$ that knows of the secret key (x, x') in Protocol 2 requires a total of $\Omega(N^{5/3})$ quantum queries to functions f and t. Besides, any strategy $\mathcal{A}$ asking $o(N^{5/3})$ queries can find this secret key* only *with $o(1)$ probability over the random views of the protocol.*

The proof of this theorem is also a four-step procedure that follows the same lines as the lower-bound proof in Section 4.1.2 with several modifications. The main change is that in the first step we compose the 2XOR problem, instead of element distinctness, with N instances of problem pSEARCH, thus defining a search problem $\mathsf{H} = \mathsf{2XOR} \circ \mathsf{pSEARCH}^N$ related to the hardness of breaking our protocol. The other three steps remain the same, except for technical adjustments taking into account 2XOR. However, there is only one *new ingredient.* We prove the optimal lower bound of 2XOR, see Section 4.2.3. This step is necessary to invoke the composition theorem as explained in Section 5.3.3.

For the first step, consider a function $\xi : [N] \to [M]$ such that there exists a (single) pair (i, j) with $\xi(i) \oplus \xi(j) = w$ and $1 \leqslant i < j \leqslant N$. The problem is to find this pair, which is a variant of 2XOR (see Definition 2.8.9).

Ambainis' algorithm for element distinctness [4] can find this pair with $O(N^{2/3})$ queries to function ξ. Besides, we reduce element distinctness to 2XOR. Therefore, finding such pair (i, j) requires $\Theta(N^{2/3})$ quantum queries. The full proof of the quantum query complexity of 2XOR is explained in Section 4.2.3.

Consider now a function $h : [N] \times [N^2] \to [M]'$. The domain of this function is composed of N "buckets" of size N^2, where $h(i, \cdot)$ corresponds to the i^{th} bucket, $1 \leqslant i \leqslant N$. In bucket i, all values of the function are 0 except for *one single* random

$x_i \in [N^2]$ for which $h(i, x_i) = \xi(i)$:

$$h(i,j) = \begin{cases} \xi(i) & \text{if } j = x_i \\ 0 & \text{otherwise}\,. \end{cases}$$

It follows from the definitions of ξ and h that there is a single pair of distinct a and b in the domain of h such that $h(a) \oplus h(b) = w$ and $h(a) \neq 0$ and $h(b) \neq 0$. How difficult is it to find this pair given a black-box function h but no direct access to ξ?

Lemma 3 (Lower bound for h)**.** *Given h structured as above, finding the pair of distinct elements a and b in the domain of h such that $h(a) \oplus h(b) = w$ and $h(a) \neq 0$ and $h(b) \neq 0$ requires $\Omega(N^{5/3})$ quantum queries to h. Any strategy making $o(N^{5/3})$ queries solves this problem only with $o(1)$ probability over the randomness of the considered oracles.*

Proof. The problem can be modelled as the composition of 2XOR across buckets with finding the single non-zero entry in each bucket. More precisely, it is a problem of *searching* among N^2 possibilities for the single non-zero $h(i, \cdot)$ for each i and then finding two of *those elements*, among N possibilities, whose *exclusive-or* equals w.

For the same reason mentioned in the previous protocol, both composition theorem [56, 81] are equally not applicable in our case because the *inner* function is not Boolean. Therefore, we use the more general composition theorem [29] again. In particular, this problem becomes a special case of technical Theorem 2.13.8 with parameters $\kappa = N$ (the number of buckets) and $\eta = N^2$ (the size of the buckets). Using Theorem 2.13.6, it follows that finding the desired pair (a, b) requires

$$\Omega(\kappa^{2/3}\eta^{1/2}) = \Omega(N^{2/3}\sqrt{N^2}\,) = \Omega(N^{5/3})$$

quantum queries to h, except with vanishing probability. $\square$

For Step 3, consider a slightly less structured search problem in which there are no longer buckets, but again with an added coordinate in the image of the function:

$$h' : [N^3] \to [N]' \times [M]'$$

There is also another justification of the added coordinate $[N]'$ that we discuss in the last step of the proof where it turns out to be necessary. This function is defined such that $h'(a) = (0, 0)$ on all but N randomly chosen points in its domain, namely w_1, w_2,…, w_N. On these N points, $h'(w_i) = (i, \xi(i))$, where ξ is the function for 2XOR considered at the beginning of the first step. We are required to find the unique pair of distinct a and b in $[N^3]$ such that $\pi_2(h'(a)) \oplus \pi_2(h'(b)) = w$ and $\pi_2(h'(a)) \neq 0$ and $\pi_2(h'(b)) \neq 0$.

Similarly to the argument in Section 4.1.2, the lower bound on the earlier search problem concerning h implies directly the same lower bound on the new search problem concerning h'. The next Lemma formalizes this argument.

Lemma 4 (Lower bound for h')**.** *Given h' structured as above, finding the pair of distinct elements a and b in the domain of h' such that $\pi_2(h'(a)) \oplus \pi_2(h'(b)) = w$ and $\pi_2(h'(a)) \neq 0$ and $\pi_2(h'(b)) \neq 0$ requires $\Omega(N^{5/3})$ quantum queries to h', except with vanishing probability over the considered oracles.*

Proof. Define intermediary function $\tilde{h} : [N] \times [N^2] \to [M]' \times [M]'$ by

$$\tilde{h}(i,j) = \begin{cases} (i, h(i,j)) & = & (i, \xi(i)) & \text{if } h(i,j) \neq 0 \\ (0, h(i,j)) & = & (0,0) & \text{otherwise}\,. \end{cases}$$

As mentioned in Section 4.1.2 at this stage, we discuss now how to reduce the search problem concerning h to the one concerning $\tilde{h}$ as well as the search problem concerning $\tilde{h}$ to the one concerning h'.

For the first reduction let $\mathcal{B}$ be an algorithm that solves $\tilde{h}$ and derive an algorithm $\mathcal{B}'$ to compute h using $\mathcal{B}$ as subroutine, and oracle access to h but not $\tilde{h}$. When $\mathcal{B}$ queries $\tilde{h}(i,j)$ then algorithm $\mathcal{B}'$ transforms an instance of h to an instance of $\tilde{h}$ as follows:

– queries $h(i,j)$

– If $h(i,j) = 0$, return $(0,0)$ as value of $\tilde{h}(i,j)$.

– Otherwise, return $(i, h(i,j)))$ as value of $\tilde{h}(i,j)$.

For the second reduction, we define algorithms $\mathcal{B}$ for h' and $\mathcal{B}'$ for $\tilde{h}$ similarly. When $\mathcal{B}$ queries $h'(k)$, then algorithm $\mathcal{B}'$ is derived using oracle $\tilde{h}$ as follows:

– transforms k into $(i,j) \in [N] \times [N^2]$ using any canonical pairing function;

– If $\tilde{h}(i,j) = (0,0)$, return $(0,0)$ as value of $h(k)$.

– Otherwise, return $(i, \tilde{h}(i,j))$ as value of $h(k)$.

Therefore, the lower bound concerning h given by Lemma 9 applies *mutatis mutandis* to h'. □

Finally, it remains to achieve Step 4 of Theorem 4.2.2.

Proof of Theorem 4.2.2. Consider any eavesdropping strategy $\mathcal{A}$ that listens to the communication between Alice and Bob and tries to determine the key (x, x') by querying black-box functions f and t. Using a function $h' : [N^3] \to [N]' \times [M]'$ as described before, we want to simulate a random run of the protocol. Equivalently, we want to solve the search problem by using unsuspecting $\mathcal{A}$ as a resource.

We supply $\mathcal{A}$ with a fake "conversation" between "Alice" and "Bob" as follows. For sufficiently large k and k', we choose randomly N points y_1, y_2,…, y_N in $[N^k]$ and one point $w \in [N^{k'}]$ and we pretend that Alice has sent the y's to Bob and that Bob has responded with w. We also choose random functions $\hat{f} : [N^3] \to [N^k]$ and $\hat{t} : [N^3] \to [N^{k'}]$. The selection of $\hat{f}$ and $\hat{t}$ may take a lot of *time*, but this does not matter in query complexity (see Section 4.1.2).

Now, we wait for $\mathcal{A}$'s queries to f and t. When $\mathcal{A}$ asks for some query $i \in [N^3]$, there are two possibilities.

– If $h'(i) = (0,0)$, return $\hat{f}(i)$ and $\hat{t}(i)$ as values for $f(i)$ and $t(i)$, respectively.

– Otherwise, return $y_{\pi_1(h'(i))}$ and $\pi_2(h'(i))$ to $\mathcal{A}$ as values for $f(i)$ and $t(i)$, respectively.

Now, it is convenient to explain the utility of the left-hand coordinate in the image of h'. Whenever $h'(i) \neq (0,0)$, the algorithm $\mathcal{A}$ should get one of the points y_1, y_2,..., y_N. Without the added coordinate, one would have a value $\xi(i)$ in $[M]$ which is usually bigger than N, and we don't see for the time of writing how to map it *one-to-one* to a value in $[N]$, which can be used as index for some y. If $\mathcal{A}$ was classical, one would simply solve this problem using a table that keeps track of any $h'(i) \neq (0,0)$.

Continuing the last step, suppose $\mathcal{A}$ returns correctly the pair (i,j) for which it was told that $t(i) \oplus t(j) = w$. This pair is in fact the answer to the search problem concerning h' since $t(i) \oplus t(j) = w$ implies that $\pi_2(h'(a)) \oplus \pi_2(h'(b)) = w$ and $\pi_2(h'(a)) \neq 0$ and $\pi_2(h'(b)) \neq 0$, except with the vanishing probability that $\hat{t}(i') \oplus \hat{t}(j') = w$ for some queries i', j' that $\mathcal{A}$ asks about t.

Queries asked by $\mathcal{A}$ concerning f and t are answered in the same way as they would be if f and t were two random functions consistent with the Y and w announced by Alice and Bob during the execution of a real protocol. Indeed, Y (subset of $[N^k]$) and w (element of $[N^{k'}]$) are uniformly picked at random in both the simulated and the real worlds. Moreover, the simulated function f is such that $f(i)$ is random when $h'(i) = (0,0)$. The remaining N values are in Y, as expected by $\mathcal{A}$. On the other hand, the simulated function t is random everywhere except for one single input pair (i,j), $i < j$, for which $\hat{t}(i) \oplus \hat{t}(j) = w$, as also expected by $\mathcal{A}$. Therefore, $\mathcal{A}$ will behave in the environment provided by the simulation exactly as in the real world. Since we disregard the vanishing possibility that t might not be one-to-one, the reduction solves the search problem concerning h' whenever $\mathcal{A}$ succeeds in finding the key.

It follows that any successful cryptanalytic strategy that makes $o(N^{5/3})$ total queries to f and t would solve the search problem with only $o(N^{5/3})$ queries to function h', which is impossible, except with vanishing probability. $\square$

4.2.3 Quantum Query Complexity of 2XOR

We prove in this section the quantum query complexity of 2XOR, using the optimal bound of element distinctness and some probabilistic reduction. The lower bound of 2XOR is necessary to prove the security of our protocols. This result is important in its own. To the best of our knowledge it was not known before.

Theorem 4.2.3. *Consider a black-box function $\xi : [N] \to [M]$ and some $w \in [M]$. The problem is to find a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $\xi(i) \oplus \xi(j) = w$, or return $\nexists$ otherwise. Then, any bounded-error quantum query algorithm for the (search variant of the)* 2XOR *problem must query the oracle function $\Theta(N^{2/3})$ times.*

We proceed with the proof by first proving the upper bound, then its matching lower bound, using two separate lemmas. For the purpose of these lemmas, consider $e : [N] \to [M]$ so that there (might) exist a pair (i, j), $1 \leqslant i < j \leqslant N$, for which $e(i) = e(j)$. Ambainis' quantum algorithm for element distinctness [4] or its generalization [36] can find such pair with $O(N^{2/3})$ queries to function e and Aaronson and Shi proved that this is optimal even for the decision version of this problem [2]. In 2012, Belovs [13] proved the same lower bound using the negative adversary method, giving explicitly the adversary matrix.

Lemma 5 (Upper bound)**.** *The element distinctness algorithm [4] or subset-finding algorithm [36] solves the* 2XOR *problem in $O(N^{2/3})$ queries to the input of size N.*

Proof. We apply Ambainis' algorithm [4] with one modification: instead of looking for i and j such that $e(i) = e(j)$, we are looking for i and j such that $\xi(i) \oplus \xi(j) = w$, which is the *property* when viewed as subset finding problem (see Section 2.8.3).

The algorithm uses quantum walks on a Johnson graph—see Sect. 2.12.4 for a review of this topic. Each node of the graph contains some number r (to be determined later) of distinct elements of $[N]$, in addition to their corresponding images under ξ. We are looking for a vertex that contains the pair (i, j) such that $\xi(i) \oplus \xi(j) = w$.

We apply Theorem 2.12.2 to analyse the cost of a quantum walk on this graph. The set up cost S corresponds to querying ξ on r random elements in $[N]$, which is $\mathsf{S} = r$ queries, since we get each random value in the image of ξ with a single query. The update cost U corresponds to adding one random image of ξ not already in the node, thus $\mathsf{U} = 1$ query. The checking cost C requires us to check if there is a pair (i, j) of elements in the node such that $\xi(i) \oplus \xi(j) = w$, which can be done without any additional queries, since all necessary information are already in the node, thus $\mathsf{C} = 0$.

Putting it all together, the expected cryptanalytic cost is in the order of

$$\begin{aligned} & \mathsf{S} + \left(\tfrac{N}{r}(\sqrt{r}\,\mathsf{U} + \mathsf{C})\right) \\ = \;& \mathsf{S} + \left(\tfrac{N}{r}(\sqrt{r}\,\mathsf{U})\right) \\ = \;& \left(r + \tfrac{N}{\sqrt{r}}\right) \text{ queries to } \xi \end{aligned}$$

To minimize the number of queries to ξ, we choose r so that $r = N/\sqrt{r}$, which is $r = N^{2/3}$. It follows that a quantum algorithm is able to find the pair (i, j) with an expected $O(N^{2/3})$ queries to ξ. □

Lemma 6 (Lower bound)**.** *There exists a probabilistic reduction from element distinctness to* 2XOR*.*

Before proceeding with the lemma, we make the following important reminder. Aaronson and Shi proved the optimal lower bound of element distinctness using the polynomial method, specifically when the range M of function e is such that $M \geqslant 3N/2$. However, the lower bound becomes $\Omega(N^{1/2})$ queries whenever $M = N$. Fortunately, Ambainis [6] proved that any symmetric problem defined on some function $e : [N] \to [M]$, its polynomial degree is the same for any $M \geqslant N$. More precisley, the quantum query lower bound of a symmetric function of a large range M, which is proved using the polynomial method, implies immediately the same lower bound for any $M \geqslant N$. A function is *symmetric* if the output of the algorithm computing it remains the same even if we permute its input and/or its output.

Fortunately, element distinctness and $\mathsf{2XOR}$ are symmetric. Therefore, their query complexities remain the same for any M satisfying $M \geqslant N$. Now, it is time to reduce element distinctness to $\mathsf{2XOR}$.

Proof. We prove that any algorithm that would solve $\mathsf{2XOR}$ could be turned into one to solve element distinctness. For this purpose, let $\mathcal{A}$ be an algorithm for $\mathsf{2XOR}$ and derive an algorithm $\mathcal{A}'$ for element distinctness.

Given a uniformly $w \in [M]$ and an oracle e for element distinctness; we may assume that w is always at position 0 of the oracle. The reduction $\mathcal{A}'$ will proceed as follows:

– choose randomly $N/2$ inputs of function e.

– on query $i \in [N]$, return $e(i) \oplus w$ if i belongs to those inputs and $e(i)$ otherwise.

– when running $\mathcal{A}$ on the modified oracle, output the pair (i, j) if it is found and verified, otherwise $\perp$.

Each run of algorithm $\mathcal{A}$ would output the correct result with probability $1/2$ since w will be added to either $e(i)$ or $e(j)$ with probability $1/2$. However, the error probability can be made arbitrarily small by running $\mathcal{A}$ a constant number of times. More precisely, run the algorithm m times. If a pair (i, j) is found, then the answer is always correct, otherwise the answer is correct with probability at least $1 - 1/2^m$; classical probabilistic algorithms are reviewed in Section 2.10.

Therefore, $\mathsf{2XOR}$ requires $\Theta(N^{2/3})$ quantum queries because we have just proved an $\Omega(N^{2/3})$ lower bound matching the previous upper bound.

Note that the same proof goes through for the decision version of this problem. $\square$

CHAPTER 5

CLASSICAL PROTOCOLS AGAINST QUANTUM ADVERSARIES

We revert in this chapter to the original setting considered by Merkle in the sense that Alice and Bob are now restricted to use classical computers. Keep in mind that their unique *channel of communication is classical*, thus ruling out the benefit of any quantum communication. On the opposite side, an eavesdropper Eave is assumed to know of all the communicated messages on the public authenticated channel, and have access to unrestricted quantum computation resources. We refer to this adversarial scenario as the *classical setting*.

In a classical world where no quantum theory is mindful of, it is necessary and sufficient for *any eavesdropper* to ask $\Theta(N^2)$ queries in order to know the key established using Merkle's scheme while legitimate parties make $O(N)$ queries. In a quantum world, however, there is a quantum attack resorting directly to Grover's algorithm [46] that enables to learn the secret key in $O(N)$ queries, hence making this scheme useless from security standpoint. This naturally raises the following question: *Is any security possible at all in the classical setting?*

The most difficult part of this question was to decide which direction to take. In addition, the intuition inspired from classical results was misleading after Barak and Mahmoudy-Ghidari [8] proved that every key agreement protocol in the random oracle model in which legitimate parties make $O(N)$ queries can be broken in $O(N^2)$ queries. Indeed, considering this latter and the quadratic speed-up provided by Grover's algorithm in several problems, it becomes tempting to think that every key agreement protocol in the random oracle model can be broken in $O(N)$ quantum queries. We prove in this chapter that this intuition is wrong by exhibiting the first classical protocol provably secure against quantum adversaries, thus closing the above open problem and opening the question: *Can we do better?* Besides, we answer positively this latter question by giving a more secure protocol.

5.1 The $\Theta(N^{13/12})$ Classical Protocol

Compare with our quantum protocol described in Chapter 4. First, the considered random functions f and g are now defined on a smaller domain to compensate for the fact that classical Alice and Bob cannot use Grover's algorithm anymore. Specifically, we choose $f : [N^2] \to [N^k]$ and $g : [N^2] \times [N^2] \to [N^{k'}]$ again with sufficiently large k and k' so that they are one-to-one (no collisions in their images), except with polynomially vanishing probability. Taking into account the birthday problem (see Section 2.8.6), this condition is satisfied whenever the range size is quadratically more than the domain size, that is, for any $k > 4$ and any $k' > 8$ as calculated in Section 5.3.4. Unambiguously, k and k' are independent from those in the previous chapter. Please refer to Section 5.3.1 to see why we use the notion vanishing instead of negligible. The second difference is that Bob finds the elements using a classical probabilistic algorithm stemmed from the birthday problem instead of using Grover's search, which cannot be used in this setting. Third, only the first step remains the same. In fact, this step is unchanged in all protocols throughout this work.

As mentioned in the previous chapters, we consider the *query complexity*: In our analyses of efficiency and lower bounds, we count only the number of queries to black-box functions or, equivalently, to the underlying binary random oracle. From the latter, all our results are implicitly stated *up to logarithmic factors.*

Protocol 3 (Classical parties vs quantum adversaries)**.**

1. *Alice picks at random N distinct points, $x_1, x_2, \ldots, x_N$, in the domain of f and transmits them encrypted, by querying the black-box function f, to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$ be Alice's secret, and $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$ be the corresponding set of encrypted points.*
2. *Bob finds the pre-images x and x' of two distinct random elements in Y. To find each one of them, he chooses random distinct values in $[N^2]$ and queries f on them until one is found whose image is in Y. He is expected to succeed with probability almost* one *in $O(N^2/N) = O(N)$ queries to oracle f.*

3. *Bob sends back $w = g(x, x')$ to Alice such that $x < x'$. In addition, he chooses $\sqrt{N} - 2$ random elements from $Y \setminus \{f(x), f(x')\}$ and he forms a set Y' of cardinality $\sqrt{N}$ by adding $f(x)$ and $f(x')$ to those elements. He sends the elements of Y' to Alice in increasing order of values.*
4. *Because Alice had kept her randomly chosen set X, she knows the preimages of each element of Y'. Let X' denote $\{x \in X \mid f(x) \in Y'\}$. By exhaustive search over all pairs of elements of X', Alice can find the unique pair (x, x') such that $g(x, x') = w$. The key established by Alice and Bob is the pair (x, x').*

We analyze the query complexity of Alice and Bob to agree on a secret key. Alice makes N queries to f in Step 1 and at most N queries to g in Step 4 because there are $\sqrt{N}\sqrt{N} = N$ pairs of elements of X' and one of them is the correct one. As for Bob, he makes an expected $O(N)$ queries to f in Step 2 and a singe query to g in Step 3. Indeed, since the domain of f contains N^2 elements, he can invert an element in Y with probability (close to 1) after $O(N) = O(\sqrt{N^2}\,)$ queries using probabilistic arguments. His time for sorting and binary search is neglected since the oracle is not involved.

Therefore, the total expected number of queries to f and g is therefore in $O(N)$ for both legitimate parties. If the protocol is constructed over a binary random oracle, it will have to be called $O(N \log N)$ times since it takes $O(\log N)$ binary queries to compute either function on any given input (see Section 4.1).

5.1.1 Quantum Attack

Again all the cryptanalytic attacks against this scheme such as direct use of Grover's algorithm, generalized Grover's algorithm, or amplitude amplification [26] require of the eavesdropper $\Omega(N^{5/4})$ queries to functions f and/or g. For a review of these essential search tools, we refer the reader to Section 2.12.

However, the same powerful attack used in Section 4.1.1 allows the eavesdropper to learn Alice and Bob's key (x, x') with an expected $O(N^{13/12}\sqrt{\log N})$ quantum queries to f and $O(\sqrt{N})$ quantum queries to g.

Theorem 5.1.1. *There exists an eavesdropping strategy that outputs the pair* (x, x') *in Protocol 3 with* $O(N^{13/12}\sqrt{\log N})$ *expected quantum queries to functions* f *and* g.

Proof. An eavesdropper proceeds with quantum walks in a graph very similar to the one explained in Section 4.1.1, except that now the vertices in the graph contain r distinct elements of X' (rather than of X) and the function's domain size is N^2 rather than N^3. In this case, the eavesdropper can find random elements of X' from his knowledge of Y' with an expected

$$O\left(\sqrt{N^2/\sqrt{N}}\right) = O\left(N^{3/4}\right)$$

queries to f per element of X', using generalized Grover's algorithm. Therefore, the set up cost is $\mathsf{S} = O(rN^{3/4})$ queries to f, which corresponds to find r elements in X'. The update cost is $\mathsf{U} = O(N^{3/4})$ queries to f, which corresponds to find one element in X' not already in the node. The checking cost is $\mathsf{C} = O(r)$ queries to g using Grover's algorithm, which requires to decide if there is a pair (x, x') in the node such that $g(x, x') = w$, which is the value sent from Bob to Alice. Finally, the eigenvalue gap δ remains in $\Omega(1/r \log r)$ but ε changes into $\Omega(r^2/N)$.

Putting all necessary ingredients together, and using the MNRS theorem, the expected cryptanalytic cost is in the order of:

$$\begin{aligned}
& \mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right) \\
= \quad & \mathsf{S} + \tfrac{\sqrt{N}}{r}\left(\sqrt{r\log r}\,\mathsf{U} + \mathsf{C}\right) \\
= \quad & \left(rN^{3/4} \text{ queries to } f\right) + \tfrac{\sqrt{N}}{r}\left(\sqrt{r\log r}(N^{3/4} \text{ queries to } f) + (r \text{ queries to } g)\right) \\
= \quad & \left(rN^{3/4} + \frac{N^{5/4}}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } f \ \ \textbf{and} \ \ \sqrt{N} \text{ queries to } g\,.
\end{aligned}$$

To minimize the number of queries to f, we choose r so that $rN^{3/4} = N^{5/4}/\sqrt{r}$, which is $r = N^{1/3}$. It follows that a quantum eavesdropper finds the key (x, x') with an expected $O(N^{13/12}\sqrt{\log N})$ queries to f and $O(\sqrt{N}\,)$ queries to g. $\square$

Note that the use of Grover's algorithm in the checking phase was not necessary to prove the theorem. Should this step be carried out classically, this would result in $\mathsf{C} = O(r^2)$ queries to g or $O(N^{5/6})$ queries to g in total.

5.1.2 Lower Bound

The proof that it is impossible to find the key (x, x') with fewer than $\Omega(N^{13/12})$ quantum queries to f and/or g, except with vanishing probability, follows the same lines as the lower bound proofs in Chapter 4 with one main difference in the fourth step.

Theorem 5.1.2 (Eavesdropping lower bound). *Any eavesdropping strategy $\mathcal{A}$ that knows of the key (x, x') in Protocol 3 requires a total of $\Omega(N^{13/12})$ quantum queries to functions f and g. Besides, any strategy $\mathcal{A}$ asking $o(N^{13/12})$ queries can find the key* only *with $o(1)$ probability over the random views of the protocol.*

Views are described in Section 2.6.1 and finding cryptographic lower bounds using reduction approach is reviewed in Section 2.14. For the reader's convenience, we start with a brief recall of the proof steps of this theorem.

1. We compose the element distinctness problem (ED) with N instances of $\mathsf{pSEARCH}$ to obtain a search problem $\mathsf{H} = \mathsf{ED} \circ \mathsf{pSEARCH}^N$.
2. We prove a lower bound on the difficulty to solve H (Lemma 7);
3. We reduce H to a less structured problem H' (Lemma 8); and
4. We reduce H' to the eavesdropping problem against our protocol.

For the first step, consider a function $\xi : [\sqrt{N}\,] \rightarrow [\sqrt{N}\,]$ such that there is a single pair (i, j), $1 \leqslant i < j \leqslant \sqrt{N}$, for which $\xi(i) = \xi(j)$, which is (a variant) of the element distinctness problem defined on a smaller domain. Ambainis' algorithm [4] can find this pair with $\Omega((\sqrt{N}\,)^{2/3}) = \Omega(N^{1/3})$ queries to function ξ and Aaronson and Shi proved that this is optimal even for the decision version of this problem [2]. Before proceeding, recall that $[K]'$ denotes $\{0\} \cup [K]$ for any natural number K.

Now, consider a function $h : [\sqrt{N}\,] \times [N^{3/2}] \to [\sqrt{N}\,]'$ where $h(i, \cdot)$ denotes the i^{th} bucket, $1 \leqslant i \leqslant \sqrt{N}$. In bucket i, all values of the function are 0 except for one. There is a single random $x_i \in [N^{3/2}]$ such that $h(i, x_i) = \xi(i)$. Symbolically,

$$h(i,j) = \begin{cases} \xi(i) & \text{if } j = x_i \\ 0 & \text{otherwise}\,. \end{cases}$$

From the definitions of ξ and h follows that there is a single pair of distinct a and b in the domain of h such that $h(a) = h(b) \neq 0$. Given an oracle access for h but *no direct access* to ξ, the query complexity of this problem is given by the following lemma.

Lemma 7. *Given h structured as above, finding the pair of distinct elements a and b in the domain of h such that $h(a) = h(b) \neq 0$ requires $\Omega(N^{13/12})$ quantum queries to h. Besides, any algorithm $\mathcal{A}$ making $o(N^{13/12})$ queries can solve this problem* only *with $o(1)$ probability over the coin tosses of $\mathcal{A}$ and the random oracles.*

Proof. The proof is similar to the one for Lemma 1, *mutatis mutandis.* The search problem is a composition of element distinctness across buckets with finding the single non-zero entry in each bucket. More precisely, it is a problem of *searching* among $N^{3/2}$ possibilities for the unique non-zero $h(i, \cdot)$ for each i, and then *finding two* of those $\sqrt{N}$ elements that are *equal.* It is a special case of Theorem 2.13.8, but with parameters $\kappa = \sqrt{N}$ (the number of buckets) and $\eta = N^{3/2}$ (the size of the buckets). It follows that finding the desired pair (a, b) requires

$$\Omega(\kappa^{2/3}\eta^{1/2}) = \Omega\Big(\sqrt{N}^{2/3}\sqrt{N^{3/2}}\Big) = \Omega(N^{13/12})$$

quantum queries to h, except with vanishing probability. □

Let $h' : [N^2] \to [\sqrt{N}\,]' \times [\sqrt{N}\,]'$ denote the less structured version of the same search problem for h, defined the same way as in Section 4.1.2, *mutatis mutandis.* There is a single pair of distinct elements a and b such that $\pi_2(h'(a)) = \pi_2(h'(b)) \neq 0$.

Finding this pair is at least as difficult as finding the collision in h. This argument is formalized by the following lemma whose proof is the same as that in Section 4.1.2.

Lemma 8. *Given h' structured as above, finding the pair of distinct elements a and b in the domain of h' such that $\pi_2(h'(a)) = \pi_2(h'(b)) \neq 0$ requires $\Omega(N^{13/12})$ quantum queries to h', except with vanishing probability.*

Finally, it remains to show that the search problem concerning h' reduces to the cryptanalytic difficulty for the eavesdropper against the running protocol.

Proof. Consider any eavesdropping strategy $\mathcal{A}$ that listens to the communication between Alice and Bob and tries to learn the key (x, x') by querying f and g. The reduction does not have direct access to machines Alice and Bob, but rather to $h' : [N^2] \to [\sqrt{N}\,]' \times [\sqrt{N}\,]'$ as described above for which we want to solve the search problem using $\mathcal{A}$ as a subroutine.

We choose random functions $\hat{f} : [N^2] \to [N^k]$ and $\hat{g} : [N^2] \times [N^2] \to [N^{k'}]$ as well as a random Boolean $s \in \{\mathsf{true}, \mathsf{false}\}$ which has the same purpose as in the proof of Theorem 4.1.2. Let $\mathrm{Im}(\hat{f})$ denote the image of function $\hat{f}$. We then supply $\mathcal{A}$ with a fake "conversation" between "Alice" and "Bob" as follows. We choose randomly $\sqrt{N}$ points y'_1, y'_2,..., $y'_{\sqrt{N}}$ in $[N^k]$, denoted by the subset Y' say, $N - \sqrt{N}$ points $y_1, y_2, \ldots, y_{N-\sqrt{N}}$ in $\mathrm{Im}(\hat{f})$, denoted by Y'', and one point $w \in [N^{k'}]$. We pretend that Alice has sent the list $Y' \cup Y''$ to Bob (in random order) and that Bob has responded with Y' in increasing order, and w.

We explain why Y' and Y'' should be sent in two different orders. If Bob sends Y' to Alice in the same order she received, then this would imply that (at least) the last element in Y', that is $y'_{\sqrt{N}}$, contains the first half of the secret, say. If this is the occurrence, a quantum eavesdropper proceeds as follows. Using Grover's algorithm, he first inverts $y'_{\sqrt{N}}$, which can be done in $O(N)$ queries. Let x denote this part of the secret. Using x, he can invert $g(x, x')$ also using Grover's algorithm, which can be done again in $O(N)$ queries. Note that Grover's search space is $[N^2]$ in both steps. Continuing Step 4, the reduction using $\mathcal{A}$ as a subroutine is derived as follows.

We wait for $\mathcal{A}$'s queries to f and g.

- When $\mathcal{A}$ asks for $f(i)$ for some $i \in [N^2]$, there are two possibilities:
  - If $h'(i) = (0, 0)$, return $\hat{f}(i)$ to $\mathcal{A}$ as value for $f(i)$.
  - Otherwise, return $y'_{\pi_1(h'(i))}$.
- When $\mathcal{A}$ asks for $g(i, j)$ for some $i, j \in [N^2]$, there are two possibilities:
  - If $\pi_2(h'(i)) = \pi_2(h'(j)) \neq 0$ and either s is true and $i < j$ or s is false and $i > j$, return w as value for $g(i, j)$.
  - Otherwise, return $\hat{g}(i, j)$.

Suppose $\mathcal{A}$ correctly returns the pair (i, j) for which it was told that $g(i, j) = w$, which is what a successful eavesdropper is supposed to do. This pair is in fact the answer to the search problem concerning function h'. Indeed $g(i, j) = w$ only for the pair (i, j) for which $\pi_2(h'(i)) = \pi_2(h'(j)) \neq 0$, except with the polynomially vanishing probability that $\hat{g}(i', j') = w$ for some query (i', j') that $\mathcal{A}$ asks about g. Actually, this event happens with a vanishing probability $N^4/N^{k'}$ (Section 5.3.4).

To create an environment identical to the real one we need an additional condition: if $y \in Y''$ then $h'(f^{-1}(y)) = (0, 0)$. This is required for all elements in Y'' to be accessible when $\mathcal{A}$ is querying f in the reduction. Except with vanishing probability, this condition is easily satisfied when k is large enough, which is the case here.

Provided the condition under discussion on Y'' is satisfied, queries asked by $\mathcal{A}$ concerning f and g are answered in the same way as they would be if both f and g were random functions consistent with the Y', Y'' and w announced by Alice and Bob during the execution of the protocol. Indeed, remember that Y' and Y'' (subsets of $[N^k]$) and w (element of $[N^{k'}]$) are uniformly picked at random in both the simulated and the real worlds. Moreover, the simulated function f is such that $f(i)$ is random when $h'(i) = (0, 0)$. Among these $N^2 - \sqrt{N}$ input values, there are exactly $N - \sqrt{N}$ output values in Y'' as expected by $\mathcal{A}$. The remaining $\sqrt{N}$ input values $1 \leqslant i \leqslant \sqrt{N}$, also satisfy $f(i) \in Y'$ as it should be. On the other hand, the simulated function g is random everywhere except for one single pair $(i, j), i \neq j$, for which $g(i, j) = w$, as it is also expected by $\mathcal{A}$. Therefore, $\mathcal{A}$ will behave in the environment provided by the simulation exactly as in the real case. Since we disregard the vanishing possibility that g might not be one-to-one, the reduction solves the search problem concerning h' whenever $\mathcal{A}$ succeeds in finding the secret.

It follows that any successful cryptanalytic strategy that makes $o(N^{13/12})$ total queries to f and/or g would solve the search problem with only $o(N^{13/12})$ queries to function h', which is impossible by Lemma 8, except with vanishing probability. This demonstrates the $\Omega(N^{13/12})$ lower bound on the quantum eavesdropping difficulty against our classical protocol, which matches the upper bound provided explicitly in Section 5.1.1. Therefore, it is possible for classical Alice and Bob to agree on a secret after an expected number of queries in the order of N whereas it is not possible, even for a *quantum* eavesdropper, to be privy of their secret with the same query complexity, except with vanishing probability. □

For pedagogical reason we mention briefly another method to prove Step 4. The key observation is to prove that *only* X' provides useful information. Equivalently, the remaining part $X \setminus X'$, provides no more than purely random information. Consequently, the reduction becomes exactly as in the last step of Theorem 4.1.2. In fact, we proved this reduction, but we don't mention it here to maintain the flow of writing. This method provides us with the following general useful observation: in such context, providing the adversary with information independent of the secret key is not helpful at all.

In the upcoming protocol we present not only a more interesting result but also a simpler lower-bound proof.

5.2 The $\Theta(N^{7/6})$ Classical Protocol

Similarly to Protocol 2 in Chapter 4 , we assume the existence of *two* black-box functions $f : [N^2] \to [N^k]$ and $t : [N^2] \to [N^{k'}]$ that can be accessed in quantum superposition of inputs. Take $k > 4$ so that the function f is one-to-one, except with polynomially vanishing probability.

The constant k' is chosen large enough to ensure that t is one-to-one, and for any integer w, there are no distinct elements $\{a, b, c, d\}$ such $t(a) \oplus t(b) = w$ and $t(c) \oplus t(d) = w$, except with vanishing probability. The problem becomes a special case of Theorem 6.1.1. Therefore, the probability of this event is at most $N^8/N^{k'}$, which vanishes quickly for any $k' > 8$. Actually, this probability is at most $N^4/N^{k'}$, vanishing for any $k' > 4$; however, we don't bother with this issue. For simplicity, we shall systematically disregard the possibility that such exceptions might occur. We give now the protocol, whose first two steps are exactly the same as in Protocol 3.

Protocol 4 (Classical parties vs quantum adversaries)**.**

1. *Alice picks at random N distinct points, $x_1, x_2, \ldots, x_N$, with $x_i \in [N^2]$ and transmits their encrypted values $y_i = f(x_i)$ to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$ be the secret set of Alice and $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$ be the set sent to Bob.*
2. *Bob finds the pre-images x and x' of two distinct random elements in Y. To find each one of them, he chooses random distinct values in $[N^2]$ and applies f to them until one is found whose image is in Y. He is expected to succeed after $O(N)$ queries to function f.*
3. *Bob sends back $w = t(x) \oplus t(x')$ to Alice.*
4. *Alice queries the oracle t on her randomly chosen set X she has kept secret. There are only N^2 candidate pairs $(x_i, x_j) \in X \times X$ such that $t(x_i) \oplus t(x_j)$ could equal w. Therefore, she can find the unique pair (x_i, x_j) without any additional queries. The key established by Alice and Bob is (x, x') with $x < x'$.*

Clearly, the expected classical queries to f and t is in $O(N)$ for legitimate parties.

5.2.1 Quantum Attack

The same quantum attack used in Section 4.2.1, which combines the subset-finding algorithm [4, 36] with the quantum algorithm for testing group commutativity [58], allows the eavesdropper Eave to recover Alice and Bob's key (x, x') with an expected $O(N^{7/6}\sqrt{\log N})$ queries to f and $O(N^{2/3}\sqrt{\log N})$ queries to g.

Theorem 5.2.1. *There exists an eavesdropping strategy that outputs the pair (x, x') in Protocol 4 with $O(N^{7/6}\sqrt{\log N})$ expected quantum queries to functions f and t.*

Proof. A quantum eavesdropper can set up a quantum attack very similar to the one explained in Section 4.2.1, except that now the functions domain size is N^2 instead of N^3. Compared with Protocol 3, the set X' doesn't exist any more. Since Y is known, the eavesdropper can find random elements of X with an expected

$$O\left(\sqrt{N^2/N}\right) = O(\sqrt{N})$$

queries to f per element of X. Therefore, the setup cost $\mathsf{S} = O(r\sqrt{N})$ queries to f and r queries to t, the update cost $\mathsf{U} = O(\sqrt{N})$ queries to f and one query to t, and the checking cost $\mathsf{C} = 0$. Note that ε becomes in $\Omega(r^2/N^2)$ in this graph.

Putting it all together, the expected quantum cryptanalytic cost is

$$\begin{aligned} & \mathsf{S} + \tfrac{N}{r}\left(\sqrt{r\log r}\,\mathsf{U}\right) \\ = \quad & \left(r\sqrt{N} + \frac{N^{3/2}}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } f \quad \textbf{and} \quad \left(r + \frac{N}{\sqrt{r}}\sqrt{\log r}\right) \text{ queries to } t\,. \end{aligned}$$

To minimize the number of queries to f, we choose r so that $rN^{1/2} = N^{3/2}/\sqrt{r}$, which is $r = N^{2/3}$. It follows that a quantum eavesdropper can find the key (x, x') with an expected $O(N^{7/6}\sqrt{\log N})$ queries to f and $O(N^{2/3}\sqrt{\log N})$ queries to t. □

5.2.2 Lower Bound

To prove the hereunder theorem, we follow the same lines as the lower bound proof in Section 4.2.2.

Theorem 5.2.2 (Eavesdropping lower bound)**.** *Any eavesdropping strategy that learns the key (x, x') in Protocol 4 requires a total of $\Omega(N^{7/6})$ quantum queries to functions f and t. Besides, any strategy $\mathcal{A}$ making $o(N^{7/6})$ queries can find this secret key* only *with $o(1)$ probability over the random views of the protocol.*

5.3 Technical Discussions

In this section we discuss in more details some relevant issues we put then here for the readers's convenience.

5.3.1 Negligible vs Vanishing Probability of Collisions

We discuss why we are satisfied with *vanishing probability* of collisions in functions rather than *negligible probability.* The are two reasons.

First of all requiring the stronger assumption does not provide any security advantage since any quantum adversary making $O(N^2)$ queries is able to break any key agreement protocol in random oracle model up to some logarithmic factor, at the time of writing. Classically, this upper bound is optimal and super-polynomial security against quantum adversaries is impossible in this model.

The second reason is problematic in some cases, in particular, when considering binary oracle which implements a random function from integers to $\{0, 1\}$. Consider for instance a black-box function $f : [N^3] \rightarrow [N^k]$. The domain size of this function is the maximum possible such that Alice and Bob can agree on a secret even using a quantum computer. Classically, this domain must be reduced to N^2 points. Anyway, for the purpose of our argument, it does not matter as long as the domain size is polynomial in N. In order to avoid collisions in the image of f except with negligible probability, it is necessary to have a range of exponential size, taking into account the definition of negligible functions (see Definition 2.1.1). Accordingly, each image requires $poly(N)$ bits to be represented and each query $i \in [N^3]$ for f requires $poly(N)$ queries to its corresponding binary oracle to construct the integer $f(i)$. We could no longer disregard this polynomial factor in our analyses, should we have considered functions with exponential range as we could do with logarithmic factors. Therefore, it is necessary that the range size to be polynomially upper-bounded, otherwise all the lower-bound proofs fail.

5.3.2 Negligible vs Vanishing Adversarial Success Probability

In standard cryptography an efficient adversary can succeed in breaking any scheme only with *negligible probability*, which is a necessary condition to satisfy. For this, consider an efficient algorithm $\mathcal{A}$ who succeeds in breaking a scheme with non-negligible probability, that is, with probability $1/poly(N)$ for some polynomial $poly$. By repeating $\mathcal{A}$ a polynomial number of times, this probability could be amplified significantly while preserving the overall polynomial running time, since polynomials are closed under composition or multiplication. However, if the probability of success is negligible then it remains so even after running $\mathcal{A}$ as a subroutine a polynomial number of times. Therefore, in our security framework it doesn't worth to require negligible adversarial success probability because all known protocols can be broken by a polynomial adversary. In other words, such requirement does not have any security advantage.

5.3.3 Needed Optimal Bounds in Composition Theorem

In this section we explain why the lower bound of 2XOR must be optimal before using it in the composition theorem.

Since our proof of the lower bound is derived using the generalized adversary method [81], the considered problems must have adversary bounds in order to apply the composition theorem. In our context, the adversary bound of the inner function pSEARCH is proven [29], however, we do not know that of 2XOR. Here is the key idea to get around this matter.

We already know that the quantum algorithm for element distinctness [4] makes $O(N^{2/3})$ queries and this is optimal [2]. In Section 4.2.3 we proved an $\Omega(N^{2/3})$ lower bound for 2XOR using a probabilistic reduction from element distinctness. However, we cannot use it immediately as an adversary bound. Now, the quantum algorithms for element distinctness [4, 36] solve 2XOR in $O(N^{2/3})$ quantum queries. Therefore, the 2XOR problem has query complexity $\Theta(N^{2/3})$. We need one more step.

A recent theorem of Ref. [56] shows that the generalized adversary bound is tight for total and partial functions. Since we know the tight bounds of 2XOR, we conclude that there exist an $\Omega(N^{2/3})$ adversary bound for it. There is no need to find it explicitly!

5.3.4 Probability of Collisions in Oracle Functions

In this section we compute the probability that two elements are mapped to the same image under a given function $f : [D] \rightarrow [R]$ for two integers D and R. This event is known as *collision* of the function denoted by Coll. We consider two cases: the classical case and the quantum one, although the latter equally deals with the former.

First, we examine the quantum case that may be easier to understand. Since quantum queries are usually made in superposition of all possible inputs, it is mandatory to calculate the probability of a collision as if the running quantum algorithm evaluates f on all the points in the domain at once, that is, D queries. Therefore, the problem can be thought of as a variant of the birthday problem (Section 2.8.6), that we reformulate it here for our convenience.

Denoted by ε, the probability of finding at least one collision after D evaluations is estimated to be $1 - e^{\frac{-D^2}{2R}}$. In our situation we always need to learn the minimal size of the range R for which a collision occurs only with vanishing probability. Therefore, we solve R as a function of M and ε:

$$R > \frac{D^2}{2 \ln \frac{1}{1-\varepsilon}}.$$

In the black-box model, we can simply consider $R \gg D^2$ so that ε vanishes quickly. For example, the probability of a collision in $f : [N^2] \rightarrow [N^k]$ vanishes if $k > 4$. Equivalently, the probability is in $o(1)$.

Actually, we can simply take other values making the probability of collisions vanish quadratically or even faster. For $k = 6$, the probability of collision vanishes super-quadratically.

In classical computations, the shortest answer is that the previous probability estimation holds as well. For more accurate probability estimation, things require little more work because we should consider the maximum number of queries that may be made to the oracle by a classical algorithm.

For exemplification, consider a T-query classical algorithm $\mathcal{A}$ making at most T queries to f. The answer to any new (or previously unasked query) i, for $1 \leqslant i \leqslant T$, is a random number independent of all the answers to the $i-1$ previously asked queries. More formally, we fix distinct elements $x_1, x_2 \in [N^3]$ and some $y \in [N^k]$.

Since f is a random oralce function, then

$$\Pr[f(x_1) = f(x_2) = y] = 1/N^k.$$

We are interested in upper-bounding this probability rather than computing it exactly. For each query i with $1 \leqslant i \leqslant T$, the probability that it gets the same value as a previously asked query is less than T/N^k. We refer to this event by Coll_i^T while Coll is the event of a collision after T queries. Then,

$$\Pr[\mathsf{Coll}] = \Pr[\bigcup_{1 \leqslant i \leqslant T} \mathsf{Coll}_i^T] \leqslant \frac{T^2}{N^k}.$$

Thus, the probability of Coll is less than T^2/N^k.

CHAPTER 6

GENERALIZED PROTOCOLS

We presented in Chapter 4 two original protocols for secret-key agreement over a classical channel, in which legitimate parties are allowed to use quantum computers. Furthermore, we gave in Chapter 5 the first two protocols secure against quantum adversaries even when legitimate parties are restricted to use classical computing.

We wonder whether the protocols in these chapters are optimal. In this chapter, we answer this question by generalizing our protocols, both classical and quantum. In Sections 4.2 and 5.2, we described these protocols, in which Bob finds a pair of preimages (x, x') in Alice's randomly selected set X, then sends back $t(x) \oplus t(x')$ where t is some black-box (or oracle) function and $\oplus$ is the bitwise exclusive-or. These protocols can be extended straightforwardly as follows. Bob finds k elements of X, for some constant $k \geqslant 2$, and sends the $\oplus$ of their images under t to Alice.

Accordingly, we obtain sequences of classical and quantum protocols, denoted by C_k and Q_k, respectively, with the following properties. In protocol C_k, a classical Alice establishes a key with a classical Bob in $O(N)$ queries to a random oracle in such a way that the optimal quantum eavesdropping strategy requires of the eavesdropper to query the same oracle $\Theta\left(N^{\frac{1}{2}+\frac{k}{k+1}}\right)$ expected times.

In protocol Q_k, a *classical* Alice establishes a key with a quantum Bob in $O(N)$ queries to a random oracle in such a way that the optimal quantum eavesdropping strategy requires of the eavesdropper to query the same random oracle $\Theta\left(N^{1+\frac{k}{k+1}}\right)$ expected times. Note that only Bob needs to be quantum in this setting, and our attacks against both sequences are similar to those exploited in Chapters 4 and 5, but they are supplemented by new ones.

The protocols presented in Section 4.2 and Section 5.2 are particular cases of these sequences, and therefore we refer to them as C_2 and Q_2, respectively. The two sequences are based on the kXOR problem (see Section 2.8.4), which is a special case of the kSUM problem (see Section 2.8.5).

Besides, the kXOR problem is related to another family of problems known as the *k-element distinctness* or element k-distinctness problem, which is well studied in terms of quantum as well as classical query complexity [45, 57, 65]. For instance, element distinctness is exactly 2XOR with $w = 0$.

Using quantum walks on Johnson graphs, Ambainis's k-element distinctness algorithm [4] is done in $O(N^{\frac{k}{k+1}})$ queries. As proved by Childs and Eisenberg [36], this algorithm can be applied to any problem that can be reduced to subset-finding (see Section 2.8.3), in particular kXOR and kSUM.

Given a black-box function of domain size N, the subset-finding algorithm [4, 36] can find a subset of k elements having *some given property* with an expected $O(N^{k/k+1})$ quantum queries (see Section 2.12.4).

Recalling Definition 2.8.10, let $\mathbb{G}$ be a finite Abelian group and w be an arbitrary element of $\mathbb{G}$. Given a positive integer k, the kSUM problem is to decide whether an input $x = x_1 \dots x_N \in \mathbb{G}^N$ contains a subset of k elements that sum to w. Belovs and Špalek [15] proved that the quantum query complexity of kSUM is $\Omega(N^{k/k+1})$ provided $|\mathbb{G}| \geqslant N^k$ where $|\mathbb{G}|$ denotes the size of the group. Actually, this lower bound matches the upper bound that can be obtained by applying the subset finding algorithm [36]. Therefore, this latter is optimal for this problem as was expected in the paper [36].

We prove in this chapter that our generalized protocols have the aforementioned level of security, however, after modifying them slightly by switching back to kSUM, which we had already considered in our research and tried to prove its lower bound.

As we did in the previous chapters, it suffices to consider the abelian group $\mathbb{G} = \mathbb{Z}_2^\ell$ with $\oplus$ as addition modulo 2 and ℓ an integer. We also take the range of the black-box function large enough to ensure that the legitimate parties can agree on the *same* key. This is possible thanks to Theorem 6.1.1. It turns out that this condition on the range is also sufficient for the lower bound for kSUM [15] to hold. Be aware that it is understood hereinafter that all the addition operations in the upcoming protocols are done modulo 2, whether we use $+$ or $\oplus$.

6.1 Generalized Classical Protocols

Before proceeding with the protocols, we prove a theorem that is needed to prove the correctness of protocols as well as to invoke the lower bound theorem for kSUM.

Theorem 6.1.1. *Consider a black-box function $g : [N^n] \to [N^m]$, positive integers m, n, k and N, and select uniformly at random $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ from $[N^n]$. Then, for $m > 2kn$, the probability that $g(a_1) \oplus \cdots \oplus g(a_k) = g(b_1) \oplus \cdots \oplus g(b_k)$ is vanishing. Furthermore, g is also one-to-one, except with vanishing probability.*

Proof. Let Coll denotes the event that $g(a_1) \oplus \cdots \oplus g(a_k) = g(b_1) \oplus \cdots \oplus g(b_k)$ when $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ are randomly chosen from $[N^n]$. The key observation is that the event Coll requiring $2k$ elements in $[N^n]$ happens with the same probability as a collision requiring two k-tuples in a function $h : [N^n]^k \to [N^m]$. More precisely, the probability of Coll is the same as finding two elements $x, y \in [N^n]^k$ such that $h(x) = h(y) = w$, where $h(z) = g(c_1) \oplus \cdots \oplus g(c_k)$ for some $z = (c_1, \ldots, c_k) \in [N^n]$. Hence, we come back to the birthday problem (reviewed in Section 2.8.6), which would say that this event happens with probability upper-bounded by the square of the domain size divided by the range size. Equivalently, the probability of Coll is bounded above by N^{2kn}/N^m, which is in $o(1)$.

The second part is a direct consequence. For any $k \geqslant 1$, which is always the case in our context, the function g is one-to-one except with vanishing probability. □

For the purpose of classical schemes, assume the existence of two black-box functions $f : [N^2] \to [N^c]$ and $g : [N^2] \to [N^{c'}]$. The constant c is chosen exactly as in Sections 4.2 and 5.2, except that we changed its name to avoid confusion with the constant k that is used to identify sequences. Hence, we consider $c > 4$ to ensure that f is one-to-one, except with vanishing probability.

The constant c' should satisfy two requirements: it is chosen so large that the lower bound theorem for kSUM requiring $|\mathbb{G}| \geqslant N^k$ can be applied; and there is a unique solution for the kSUM problem, allowing legitimate parties to agree on the same key, except with vanishing probability. Fortunately, these two requirements can easily be met in our case. The first condition is satisfied by letting $c' \geqslant k$. To meet the second one, we choose c' such that for any $2k$ elements $a_1, \dots, a_k$ and $b_1, \dots, b_k$, the probability that $g(a_1) \oplus \dots \oplus g(a_k) = g(b_1) \oplus \dots \oplus g(b_k)$ is vanishing. Therefore, using Theorem 6.1.1, we choose $c' > 4k$ to satisfy the two requirements. Now, we are ready to present the classical protocol C_k, for any $k \geqslant 2$. Only the first step remains the same, compared with our earlier discussed protocols.

Protocol 5 (Classical parties vs quantum adversaries)**.**

1. *Alice chooses at random N distinct points, $x_1, x_2, \dots, x_N$, in the domain of function f, and transmits the encrypted set, $Y = \{f(x_i) \mid 1 \leqslant i \leqslant N\}$, to Bob. Let $X = \{x_i \mid 1 \leqslant i \leqslant N\}$.*
2. *Bob finds k distinct elements in X, denoted by $b_1, b_2, \dots, b_k$. To find each one of them, he chooses random distinct values in $[N^2]$ and queries f on them until one is found whose image is in Y. He is expected to succeed with probability arbitrary close to* one *in $O(N^2/N) = O(N)$ queries to function f.*
3. *Bob sends back $w = g(b_1) \oplus \dots \oplus g(b_k)$ to Alice.*
4. *Alice queries the oracle g on her set X. Although there are N^k candidate k-tuples such that $g(a_1) \oplus \dots \oplus g(a_k)$ could equal w, she can find the* unique *tuple without any additional queries. The common secret between Alice and Bob is $(b_1, \dots, b_k)$ where the components of this tuple are in some public order.*

Calculating the legitimate complexities, each legitimate party asks a total of $O(N)$ classical queries. Indeed, Alice asks exactly N queries to f in Step 1 and N queries to g in Step 4. Bob asks $O(N)$ queries to f in Step 2 (use probabilistic arguments to calculate this bound) and k queries to g in Step 3.

6.1.1 Quantum Attack

An eavesdropper attacking the protocol proceeds by a quantum walk algorithm, similar to that exploited in Section 4.2.1, and whose query complexity is formulated by the following theorem.

Theorem 6.1.2. *There exists a eavesdropping strategy that outputs the k-tuple $(b_1, \ldots, b_k)$ in Protocol 5 with an expected $O(N^{\frac{1}{2}+\frac{k}{k+1}}\sqrt{\log N})$ quantum queries to f and $O(N^{\frac{k}{k+1}}\sqrt{\log N})$ quantum queries to g.*

Proof. The eavesdropping strategy is very similar to the subset-finding algorithm. Indeed, it is necessary and sufficient for the adversary to find k elements of X that sum to w, which is the value sent from Bob in the last step of Protocol 5. Hence, it is exactly the kSUM problem, except that the set X is not accessible directly. Instead of being able to get randomly chosen values in the image of g with a single query per value, Eave has to get random elements of X by applying BBHT on the list Y and query g on them, which requires $O(\sqrt{N^2/N}) = O(\sqrt{N}\,)$ queries to f and *one* query to g per element. Therefore, the number of queries to f, compared to $O(N^{k/k+1})$ queries to g, will be multiplied by $O(\sqrt{N})$ at least. The algorithm also maintains a data structure to stores the values under g along with the r-tuples.

To analyse the query complexity of our quantum walk algorithm, we apply Theorem 2.12.4. The set up cost is $\mathsf{S} = O(r\sqrt{N})$ queries to f and r queries to g. The update cost is $\mathsf{U} = O(\sqrt{N})$ queries to f and *one* query to g. The checking cost C requires us to decide whether there are k elements in the vertex such that $w = g(b_1) \oplus \cdots \oplus g(b_k)$, which can be done without oracle queries, making $\mathsf{C} = 0$. The eigenvalue gap δ remains in $\Omega(1/r \log r)$ but ε changes into $\Omega(r^k/N^k)$. We can calculate ε the same way as in Section 4.2.1.

Putting it all together, the expected eavesdropping cost is in the order of

$$\begin{aligned}
&\mathsf{S} + \left(\tfrac{N^{k/2}}{r^{k/2}}(\sqrt{r\log r}\,\mathsf{U} + \mathsf{C})\right) \\
= &\left(r\sqrt{N} \text{ calls to } f + r \text{ calls to } g\right) + \tfrac{N^{k/2}}{r^{k/2}}\sqrt{r\log r}\left(\sqrt{N} \text{ calls to } f + 1 \text{ call to } g\right) \\
= &\left(r\sqrt{N} + \frac{\sqrt{N}N^{k/2}}{r^{(k-1)/2}}\sqrt{\log r}\right) \text{calls to } f \;\; \textbf{and} \;\; \left(r + \frac{N^{k/2}}{r^{(k-1)/2}}\sqrt{\log r}\right) \text{calls to } g\,.
\end{aligned}$$

Neglecting the log factor, we choose r so that $r\sqrt{N} = \sqrt{N}N^{k/2}/r^{(k-1)/2}$ to optimize the number of queries to f and g. This implies $r = N^{k/k+1}$ and the theorem follows.

□

6.1.2 Lower Bound

The lower bound on the eavesdropper's query complexity is formalized as follows.

Theorem 6.1.3 (Eavesdropping lower bound)**.** *Any eavesdropping strategy $\mathcal{A}$ that learns the key $(b_1, \ldots, b_k)$ in Protocol 5 requires a total of $\Omega(N^{\frac{1}{2}+\frac{k}{k+1}})$ quantum queries to functions f and/or g. Any strategy $\mathcal{A}$ asking $o(N^{\frac{1}{2}+\frac{k}{k+1}})$ queries can find the secret key* only *with vanishing probability over the random views of the protocol.*

The proof of this theorem follows the same lines as the lower-bound proof in Section 4.1.2. The main change is that in the first step we compose the kSUM problem, instead of element distinctness as in Protocol 1 or kXOR as in Protocol 2, with N instances of $\mathsf{pSEARCH}$. This defines the search problem $\mathsf{H} = \mathsf{kSUM} \circ \mathsf{pSEARCH}^N$.

Consider a black-box function $\xi : [N] \to [M]$ and some $w \in [M]$ such that there exists k elements $b_1, \ldots, b_k$ with $w = \xi(b_1) \oplus \cdots \oplus \xi(b_k)$ where $b_i \in [N]$ for $1 \leqslant i \leqslant N$. The problem is to find this k-subset; a variant of kSUM (Section 2.8.5). By the negative adversary method [80], Belovs and Špalek [15] proved an $\Omega(N^{k/k+1})$ quantum lower bound even for the decision version this problem. Conditions on the range size are already discussed in Chapter 4 and 5.

Consider now a function $h : [N] \times [N] \to [M]'$. The domain of this function is composed of N "buckets" of size N, where $h(i, \cdot)$ corresponds to the i^{th} bucket, $1 \leqslant i \leqslant N$. In bucket i, all values of the function are 0 except for *one single* random $x_i \in [N]$ for which $h(i, x_i) = \xi(i)$:

$$h(i,j) = \begin{cases} \xi(i) & \text{if } j = x_i \\ 0 & \text{otherwise}\,. \end{cases}$$

The definitions of kSUM and h implies that there is a k-subset of distinct elements in the domain of h such that $w = h(b_1) \oplus \cdots \oplus h(b_k)$. How difficult is it to find this k-subset given an oracle access for function h but *no* direct access to ξ?

Lemma 9 (Lower bound for h)**.** *Given h structured as above, finding a k-subset of distinct elements $\{b_1, \ldots, b_k\}$ in the domain of h such that $w = h(b_1) \oplus \cdots \oplus h(b_k)$ and $h(b_i) \neq 0$, for $1 \leqslant i \leqslant k$, requires $\Omega(N^{\frac{1}{2}+\frac{k}{k+1}})$ quantum queries to oracle h. Any strategy asking $o(N^{\frac{1}{2}+\frac{k}{k+1}})$ queries solves this problem only with $o(1)$ probability.*

Proof. The problem is then reduced to *searching* among N possibilities for the unique non-zero $h(i, \cdot)$ for each i and then finding k of *those elements*, among N possibilities, whose *sum* equals w. It is the composition of kSUM with pSEARCH. Since the lower bounds of pSEARCH and kSUM were proven [15, 29] using the generalized adversary method, we obtain the lower bound of the composed function h directly using the composition theorem with pSEARCH as inner function [29]. In particular, this problem becomes a special case of technical Theorem 2.13.8 with parameters $\kappa = N$ (the number of buckets) and $\eta = N$ (the size of the buckets). Using Theorem 2.13.6 along with the quantum query complexities for kSUM and pSEARCH, it follows that finding the desired secret requires

$$\Omega(\kappa^{k/k+1}\eta^{1/2}) = \Omega(N^{k/k+1}\sqrt{N}\,) = \Omega(N^{\frac{1}{2}+\frac{k}{k+1}})$$

quantum queries to h, except with vanishing probability. The remaining part of the proof is already explained in Section 4.1.2. □

6.2 Generalized Quantum Protocols

Similarly to classical protocols, we can generalize the quantum protocols and obtain a sequence, denoted by Q_k for $k \geqslant 2$, with the following properties. In protocol Q_k, classical (or quantum) Alice agrees on a key with a quantum Bob after $O(N)$ queries to a random oracle in such a way that the optimal quantum eavesdropping strategy requires of the eavesdropper to access the oracle $\Theta\big(N^{1+\frac{k}{k+1}}\big)$ expected times. This optimal quantum query complexity can be proven entirely the same way that we used in Section 6.1. A quantum protocol Q_k is similar to classical protocol C_k except for Step 2, which invokes BBHT, and the function domain size.

6.3 Quantum Attacks Without Logarithmic Factors

It turns out that all our quantum attacks discussed till now can be done without the square-root logarithmic factors, thanks to a recent work of Childs and Kothari [35]. Although we don't care about logarithmic factors, we describe this new attack for completeness and pedagogical purposes.

In a nutshell, consider any of the preceding attacks that we discussed. If we could have $\delta = 1/r$ instead of $1/r \log r$, then applying the same formula would give the desired result. We will describe only the attack against the classical sequence. For the quantum case, we only state the theorem since the proof is similar.

Theorem 6.3.1. *There exists an eavesdropping strategy that outputs the k-tuple in Protocol C_k with an expected $O(N^{\frac{1}{2}+\frac{k}{k+1}})$ quantum queries to functions f and g.*

Proof. This new attack combines the subset-finding quantum algorithm [4, 36] with quantum walks on a Hamming graph $H(X, r)$ whose vertex set is X^r and there is an edge between two r-tuples (vertices) if and only if they only differ on *one* coordinate. Here X is Alice's secret set and r is a parameter to be determined optimally later. The eigenvalue gap of this graph is $\delta \in \Omega(1/r)$ [35], which is the *feature* behind the removal of the logarithmic factor. As usually, the algorithm maintains a data structure. Each vertex consists of an r-tuple $|u_1 u_2 \ldots u_r\rangle$ and the corresponding images under g, where $u_i \in X$ for $1 \leqslant i \leqslant r$. We are looking for a (marked) vertex that contains k elements of X such that their sum equals w, which is the value announced by Bob in Step 3 of the protocol.

The random walk on this graph, identified by a transition matrix P, can be quantized [59, 75]. We need to know the fraction of marked vertices under the stationary distribution denoted by ε and the eigenvalue gap denoted by $\delta = \delta(P)$. Afterwards, using Theorem 2.12.4, the complexity of the resulting quantum walk algorithm is a function of three quantum costs S, U and C.

The set up cost is $\mathsf{S} = O(r\sqrt{N})$ queries to f and r queries to g. The update cost is $\mathsf{U} = O(\sqrt{N})$ queries to f and *one* query to g. The checking cost $\mathsf{C} = 0$. Following the same method in Section 4.1.1, we get $\varepsilon \approx r^k/N^k$.

Putting it all together, the expected eavesdropping cost is in the order of

$$
\begin{aligned}
& \mathsf{S} + \left(\tfrac{N^{k/2}}{r^{k/2}}(\sqrt{r}\,\mathsf{U} + \mathsf{C})\right) \\
= \;& \left(r\sqrt{N} + \frac{\sqrt{N}N^{k/2}}{r^{(k-1)/2}}\right) \text{calls to } f \;\; \textbf{and} \;\; \left(r + \frac{N^{k/2}}{r^{(k-1)/2}}\right) \text{calls to } g\,.
\end{aligned}
$$

We choose r so that $r\sqrt{N} = \sqrt{N}N^{k/2}/r^{(k-1)/2}$ to optimize the number of queries to f and g. This implies $r = N^{k/k+1}$ and the theorem follows.

Compared with the attack in Section 4.1.1, the setup phase is much simpler, being just the direct result of r independent applications of BBHT. Recall that the setup phase consists in the following two step. The changes only occur in the second step as we explain below.

1. Prepare the following state as described in Section 4.1 with domain size N^2:

$$
|\psi_1\rangle = \left(\frac{1}{\sqrt{N^2}} \sum_{u_1 \in [N^2]} |u_1\rangle\right) \otimes \cdots \otimes \left(\frac{1}{\sqrt{N^2}} \sum_{u_r \in [N^2]} |u_r\rangle\right)
$$

2. Produce the superposition state of all possible vertices of the graph:

$$
|\psi_2\rangle = \left(\frac{1}{\sqrt{N}} \sum_{u_1 \in X} |u_1\rangle\right) \otimes \cdots \otimes \left(\frac{1}{\sqrt{N}} \sum_{u_r \in X} |u_r\rangle\right)
$$

To construct $|\psi_2\rangle$ here, we proceed as follows for each register $|u_i\rangle$ for $1 \leqslant i \leqslant r$. We produce a uniform superposition of all elements $u_i \in X$ by applying BBHT on the ith register of $|\psi_1\rangle$. Since we know the exact number of solutions, N, this state can be produced with certainty in $O(\sqrt{N^2/N}) = O(\sqrt{N})$ queries even in the worst case thanks to Theorem 2.12.1. Although we applied Generalized Grover's algorithm k times, the total cost remains in $O(\sqrt{N}\,)$ queries since k is constant. $\square$

This attack also holds for any defined quantum protocol Q_k as stated formally:

Theorem 6.3.2. *There exists an eavesdropping strategy that outputs the k-tuple in Protocol Q_k with an expected $O(N^{1+\frac{k}{k+1}})$ quantum queries to functions f and g.*

It is worth mentioning that the comparison between Johnson and Hamming graphs, based on the calculations obtained from Theorem 2.12.4, reveal that the order of the coordinates has no significance on the asymptotic behaviour of the algorithm. As a result, it might be possible to transform one graph into the other. Using the expressions of Sántha and neglecting normalization, the problem can be reduced to how to get $|\phi\rangle$ from $|\psi\rangle$ where

$$|\phi\rangle = \sum_{A \subseteq X,\, |X|=r} |A\rangle$$

and

$$|\psi\rangle = \left(\sum_{x \in X} |x\rangle \right)^{\otimes r}.$$

The setup phase in a Johnson graph $J(X, r)$ is to construct $|\phi\rangle$, or equivalently, to find r random elements of X. However, considering the structure of our problem, we can construct $|\psi\rangle$, which is exactly the setup phase in a Hamming graph.

Notice finally that upper bounds can be proved using composition theorems for upper bounds [80]. In fact, we could have derived our upper bounds easily if we had sacrifice the simplicity of the protocols, which is not a good choice in cryptography.

6.4 Time Complexity

The quantum query complexity has been our measure of complexity throughout this thesis. However, this section was specifically written for those who equally care about time complexity. Needless to say, a lower bound on the query complexity of any problem is a lower bound on the time complexity of the same problem.

Even though our protocols Q_k and C_k require *classical* Alice to ask the oracle functions only $O(N)$ queries, she has to spend a *time* in $O(N^{\lceil k/2 \rceil})$ to complete the protocol using the best algorithm currently known, which is more than linear when $k \geqslant 3$. However, for $k = 2$ in the classical setting, and $k = 3$ in the quantum setting, we present procedures that achieve Alice's task in *linear time*. Depending on whether or not Alice is quantum, we consider two cases.

For these procedures, recall that Alice's set X consists of N distinct random points, $x_1, x_2, \ldots, x_N$, in the domain of f, and $Z = \{z_i = g(x_i) \mid 1 \leqslant i \leqslant N\}$ is the set that Alice obtains by querying the other function on X. Let w denote the value sent at the last step of the protocol. More precisely, let $w = z_{i_1} \cdots + z_{i_k}$ with distinct $z_{i_j} \in Z$ for $1 \leqslant j \leqslant k$. We assume available a quantum random access memory that can be accessed in superpositions (see Section 2.11.3), otherwise several important quantum algorithms [4, 20, 25, 32] won't work any more. Note that, consequently, Eave could be even less efficient against our protocols!

6.4.1 Classical Alice

When Alice is classical, a direct search approach requires time in $O(N^k)$. However, she can reduce this time to $O(N^{\lceil k/2 \rceil})$. We first start with the case $k = 2$, which is of special importance, then we treat the general case.

Theorem 6.4.1. *Given $w = g(x) \oplus g(x')$ for random elements x and x' in X, classical Alice can find this pair in time $O(N \log N)$.*

Proof. Assume that $w = z \oplus z'$ with $z, z' \in Z = \{g(x) \mid x \in X\}$ and the pair (z, z') is unique, except with vanishing probability. After receiving w, Alice builds the set $D_w = \{w \oplus g(x) \mid x \in X\}$, sorts it and searches two equal elements between D and Z. Actually, there are two solutions, whether she finds $w \oplus z = z'$ or $w \oplus z' = z$. Either solutions implies $z \oplus z' = w$, or equivalently, $g(x) \oplus g(x') = w$ for $x, x' \in X$.

It is not difficult to analyse the time complexity: there is nothing more than searching in sets of size N after sorting them. Therefore, the overall running time remains in $O(N \log N)$. $\square$

Now when $k \geqslant 2$, classical Alice can reduce the time complexity to $O(N^{\lceil k/2 \rceil})$ by a generalization of the previous algorithm, with which we proceed immediately.

Theorem 6.4.2. *Given $w = g(x_1) \oplus \cdots \oplus g(x_k)$ for random $x_1, \ldots, x_k$ in X, classical Alice can find these elements in time $O(N^{\lceil k/2 \rceil} \log N)$.*

Proof. For clarity purposes, we start with the case where k is even; let $k = 2m$ for some integer m.

1. Assume that $w = z_{i_1} \oplus \cdots \oplus z_{i_m} \oplus z_{i_{m+1}} \oplus \cdots \oplus z_{i_{2m}}$ for $z_\ell \in Z$ and $1 \leqslant \ell \leqslant 2m$.
2. Construct the table S_m containing the sums of all possible m-tuples of distinct elements of Z. Symbolically,

$$S_m = \{s_j = z_{j_1} \oplus z_{j_2} \cdots \oplus z_{j_m} \mid z_p \in Z;\ z_p \neq z_q;\ 1 \leqslant p, q \leqslant m\}.$$

   The fact that S_m is of size less than N^m has no significance on the time complexity. Note that S_m implicitly keeps track of the tuple $(z_{j_1}, \ldots, z_{j_m})$ that corresponds to the sum s_j for $1 \leqslant j \leqslant N^m$.
3. Compute a corresponding table $D_w = \{w \oplus s_j \mid 1 \leqslant j \leqslant N^m\}$ where $s_j \in S_m$. Assuming Z contains a unique k-subset $\{z_{i_1}, \ldots, z_{i_k}\}$ satisfying w, except with vanishing probability, this implies that $w \oplus (z_{i_{m+1}} \oplus \cdots \oplus z_{i_{2m}}) = z_{i_1} \oplus \cdots \oplus z_{i_m}$ and there is *at least* one common m-tuple between D_w and S_m. In fact, there are $k(k-1)\cdots(k-m+1)$ solutions. Sort D_w for the next step.
4. Search an m-subset in S_m that has a match in D_w. It takes $O(N^m \log N)$ to solve this problem; apply any classical search algorithm that finds a match between two sets. This problem can be solved even faster, having many solutions. However, it does not change the asymptotic behaviour since k is a constant.

Note that Alice does not need to store the two tables. We choose this straightforward method for simple explanation.

We analyse the time complexity of this algorithm. Step 2 takes $O(N^m)$ time to compute S_m. Step 3 takes $O(N^m \log N)$ time to prepare and sort D_w, and Step 4 is also done in $O(N^m \log N)$. Therefore, the overall running time remains in $O(N^{k/2} \log N)$.

When $k = 2m+1$, we proceed exactly the same way, except that instead of S_m we consider S_{m+1}, which is the set of all sums of $m+1$ elements of Z. $\square$

Unfortunately, for $k = 3$, the problem become 3SUM, which is long-standing in classical time complexity. The best known algorithm takes $\Theta(N^2)$ time [42]. However, we can avoid this problem by providing Alice with quantum computers (see Section 6.4.2). Recapitulating, our best protocol in the classical setting is C_2, which can be completed legitimately in linear time and query complexity while any quantum eavesdropper requires $\Omega(N^{7/6})$ quantum queries to learn the secret key.

6.4.2 Quantum Alice

Quantum Alice can achieve the last step of protocol Q_k in $O(N^{\lceil k/2 \rceil})$ time using Grover's search, which is perhaps the most straightforward quantum approach. However, she can do much better. We first start with the case $k = 3$, where Alice's time complexity is *linear*, then we treat the general case.

Theorem 6.4.3. *Given $w = g(x) \oplus g(x') \oplus g(x'')$ for random x, x' and x'' in X, quantum Alice can find these elements in time $O(N \log N)$.*

Proof. To search three elements in Z, which sum to w, Alice proceeds as follows:

1. Compute the table $D_w = \{w \oplus z_p \mid 1 \leqslant p \leqslant N; z_p \in Z\}$. We will also need $S_2 = \{z_i \oplus z_j \mid 1 \leqslant i, j \leqslant N; i \neq j\}$, which is the set of all possible sums of pairs $(z_i, z_j) \in Z \times Z$ with $z_i \neq z_j$. Assuming Z contains a unique triplet (z, z', z'') such that $w = z \oplus z' \oplus z''$, except with vanishing probability, implies that $w \oplus z'' = z \oplus z'$ and there are exactly three sums in D_w that corresponds to six possible pairs in S_2. Sort the table D_w so that one can find any of its elements in $\log N$ time, as a preparation for the next step. Then, load D_w into a quantum memory (QRAM or QROM) whose model is described in Section 2.11.3.

2. Apply BBHT to search one out of the six possible pairs (z, z') in S_2 such that $z \oplus z' \in D_w$. Since the search space is N^2, this takes $O(\sqrt{N^2/6}) = O(N)$ Grover iterations. This step illustrates the fact that a quantum memory randomly accessible in superpositions is indispensable.

3. Once a pair (z, z') is found, there is $d \in D_w$ such that $z \oplus z' = d$, implying that there exists $z'' \in Z$ such that $z \oplus z' = w \oplus z''$ or equivalently $z \oplus z' \oplus z'' = w$.

The overall running time is in $O(N \log N)$. Recapitulating, when $k = 3$ in the quantum setting, we get our best protocol that can be completed by legitimate parties in linear time and linear query complexity. Nevertheless, an adversary needs $\Omega(N^{7/4})$ quantum queries to learn the key, except with vanishing probability over the random views of the protocol. $\square$

Consider now the general case, where $k = 2m$ for some positive integer $m \geqslant 2$. The case for odd k is slightly different and we address it afterwards.

Theorem 6.4.4. *Given $w = g(x_1) \oplus \cdots \oplus g(x_k)$ for random $x_1, \ldots, x_k$ of X, quantum Alice can find this k-subset in time $O(N^{k/3} \log N)$ when k is even.*

Proof. Recall that, given two functions $F : A \to R$ and $G : B \to R$ defined on the same range, a *claw* is a pair $(a, b) \in A \times B$ such that $F(a) = G(b)$.

Alice defines two functions f and g having domain size N^m and the same range as follows:

$$f, g : \underbrace{Z \times Z \cdots Z}_{m \text{ times}} \to R$$

such that $f(u_1, \cdots, u_m) = u_1 \oplus \cdots \oplus u_m$ and $g(v_1, \cdots, v_m) = w \oplus (v_1 \oplus \cdots \oplus v_m)$ where $u_i \in Z$ and $v_i \in Z$ for $1 \leqslant i \leqslant N$. The problem is to find $2m$ elements of Z that sum to w. Thus, the problem is reduced to finding a claw in the functions, that is, two distinct m-tuples, u and v, such that $f(u) = g(v)$.

At this stage, Alice uses the subset-finding algorithm [36] (see Section 2.12.4). This process takes $S^{2/3}$ time, where $S = N^m$ is the domain size of f and g, implying $S = N^{2m/3} = N^{k/3}$. As already mentioned, there are $k(k-1)\cdots(k-m+1)$ possible solutions, reducing the time complexity even further but by a constant factor. Unfortunately, this algorithm has time complexity in $O(N^{4/3})$ when $k = 4$, which is still more than linear.

When $k = 2m + 1$, we can do the following step before applying the earlier procedure. Choose a random element $z \in Z$ and set $w' = w \oplus z$, thus transforming the current problem into the earlier one with $k' = 2m'$ and $m' = m + 1$. We can certainly use a faster algorithm avoiding this extra element. However, it is just a slight time overhead for the sake of simplicity. $\square$

Note finally that, for any $k = 3m$, there is an easy approach based on Grover's algorithm.

1. Assume that $w = z_{i_1} \oplus \cdots \oplus z_{i_m} \oplus z_{i_{m+1}} \oplus \cdots \oplus z_{i_{3m}}$ for $z_{i_j} \in Z$.
2. Construct the table S_m containing the sums of all possible m-tuples of distinct elements in Z. It has dimension $N^m = N^{k/3}$.
3. Prepare the corresponding table $D_w = \{w \oplus s_j \mid 1 \leqslant i \leqslant N^m\}$ where $s_j \in S_m$. Assuming Z contains a *unique* k-subset $\{z_{i_1}, \ldots, z_{i_k}\}$ satisfying w, this implies that $w \oplus s_j = z_{i_{m+1}} \oplus z_{i_{m+2}} \cdots \oplus z_{i_{2m}}$.
4. Define as before the subset S_{2m} having size N^{2m}. For each specific $\alpha \in S_{2m}$, we can find a matching element $\beta \in D_w$ using a classical search algorithm in the sorted table D_w. Indeed, combine this binary search in D_w with Grover's search in S_{2m}.

We analyse the time complexity of this algorithm. Step 2 takes time N^m to compute and keep the table in memory. In Step 3, we apply the addition operation and sorting on D_w, which can be done in at most $O(N^m \log N)$ time. Finally, Step 4 can be done in $O(\sqrt{N^{2m}} \log N)$ or equivalently $O(N^m \log N)$ time. Therefore, the overall running time is in $O(N^{k/3})$, when neglecting the logarithmic factor.

CHAPTER 7

CONCLUSIONS AND OPEN QUESTIONS

We studied in this thesis the problem of secret-key agreement over a *classical* communication channel in the random oracle model against quantum eavesdroppers. The major limitation was the classical channel. This exclude the benefit of transmitting quantum information, which makes quantum key distribution [16] possible. The other severe restriction emerges when legitimate parties are *not* allowed to use any computing resource beyond the classical theory.

Depending on computing resources with which legitimate parties are equipped, we considered two different settings. In the first one, which we called the classical setting, legitimate parties are restricted to using classical computers. In the second one, the quantum setting, legitimate parties have the means to make quantum computations. In both settings, the eavesdropper trying to learn the established secret key is assumed to have access to all the communicated messages and any computing strategy allowed by quantum mechanics. In all our protocols, legitimate parties query the oracle a number of times proportional to some parameter N. Our conclusions are divided into three categories: quantum setting, classical setting, and quantum computing and random oracle model.

7.1 Classical Setting

In the classical setting, which is considered in Chapter 5, we contributed the first protocol that is secure against any quantum adversary. More precisely, there is *no* quantum eavesdropping strategy able to learn the key before asking $\Omega(N^{13/12})$ quantum queries, in contrast with the common conjecture that "any key agreement protocol in the random oracle model can be broken with $O(N)$ quantum queries".

Improving on the first protocol, we gave a scheme requiring $\Omega(N^{7/6})$ quantum queries of the eavesdropper (Eave).

Furthermore, for any integer $k \geqslant 2$, we provided a classical protocol C_k with the following characteristics: a classical Alice establishes a key with a classical Bob after $O(N)$ queries to random oracles in such a way that the optimal quantum eavesdropper requires an expected $\Theta\big(N^{\frac{1}{2}+\frac{k}{k+1}}\big)$ queries, which tends to $\Theta(N^{3/2})$ when k increases.

As a result, *classical* Alice and Bob can establish a secret key against any quantum eavesdropper with as good a security (in the limit) as it was known to be possible for *quantum* Alice and Bob before this work [23].

The main open question would be to break the $\Omega(N^{3/2})$ barrier or prove that this is not possible. More precisely, the following questions come to light.

1. Is it possible to devise a classical protocol that provides exactly $\Omega(N^{3/2})$ security… or better?!
2. Can every key agreement protocol in the random oracle model be broken with $O(N^{3/2})$ quantum queries if legitimate parties remain classical?
3. What is the best possible security gap in the this restrictive setting?

For the first question, we believe that a classical protocol providing exactly $\Omega(N^{3/2})$ security *does* exist. Actually, we already have a promising one.

However, concerning the second question, we believe that every key agreement protocol in the random oracle model in which legitimate parties are classical and ask $O(N)$ queries can be broken with $O(N^{3/2})$ quantum queries, thus contrasting our point of view in the upcoming quantum case. Following our believes, the answer to Question 3 would be $\Theta(N^{3/2})$ quantum queries.

The following table compares our results with the previous ones in the classical setting (Alice and Bob are classical while Eave is always quantum).

Previous/Our schemes	Problem involved	Eave's lower bound
Merkle's scheme	OR	$\Theta(N)$
The first secure protocol	ED/SF	$\Theta(N^{13/12})$
The second secure protocol	2XOR	$\Theta(N^{7/6})$
The kth secure protocol C_k, for $k \geqslant 2$	kSUM	$\Theta(N^{\frac{1}{2}+\frac{k}{k+1}})$

7.2 Quantum Setting

In the quantum setting (see Chapter 4), we presented two (quantum) protocols improving on the first attempt [23] to repair Merkle's scheme. This latter provides optimal quadratic security in the classical world [8], however, has *no security* at all against a quantum eavesdropper applying Grover's algorithm straightforwardly. Both protocols require $\Omega(N^{5/3})$ quantum queries of Eave in order to learn the secret. They are not only better than the earlier $\Omega(N^{3/2})$ scheme [23], but again disagree with the conjecture that "any key agreement protocol in the random oracle model can be broken with $O(N^{3/2})$ quantum queries, when Alice and Bob are quantum".

Furthermore, for any integer $k \geqslant 2$, we provided a quantum protocol Q_k with the following properties: a *classical* Alice establishes a key with a quantum Bob after $O(N)$ queries to a random oracle in such a way that any quantum eavesdropper requires an expected $\Theta\big(N^{1+\frac{k}{k+1}}\big)$ queries, thus approaching $\Theta(N^2)$ when k increases. Notice that *only* Bob needs to be quantum in this setting unless we equally care about time complexity, which is discussed in Section 6.4.

Consequently, key agreement protocols inspired by Merkle can be arbitrarily as secure in our quantum world as they were in the classical computer world in 1974. More precisely, they can be arbitrarily close to quadratic security. Considering our results, we raise several open questions:

1. Can the quadratic security of Merkle's scheme be restored *exactly* rather than in the limit if all parties make use of quantum computers?
2. Can *every* key agreement protocol in the random oracle model be broken with $O(N^2)$ quantum queries when legitimate parties are quantum?
3. Is it possible to find a quantum protocol that provides better than quadratic security…?!

While we believe that a quantum protocol providing exact quadratic security *does* exist, it is difficult to speculate on the other two questions. Indeed, even though it was proven in the classical case that the optimal security is quadratic [8], there is no compelling evidence that such a limitation exists in a quantum world.

The following table compares our results with the previous ones in the quantum setting (Alice might be quantum, Bob and Eave are always quantum).

Previous/Our schemes	Problem involved	Eave's lower bound
Brassard-Salvail scheme	OR	$\Theta(N^{3/2})$
Our first scheme	ED/SF	$\Theta(N^{5/3})$
Our second sceme	2XOR	$\Theta(N^{5/3})$
Our kth protocol Q_k, for $k \geqslant 2$	kSUM	$\Theta(N^{1+\frac{k}{k+1}})$

Since we highlight our contributions in this section, it is worth mentioning that our paper at Crypto 2011 [29] was considered for a "Best Paper Award", according to an email from the Program Chair.

7.3 Quantum Computing and the Random Oracle Model

In addition to our cryptographic research topic in this thesis, we have explored some questions related to capabilities and limitations of quantum computing and the random oracle model. Considering our results, we would like to add the following observations:

The random oracle model in a quantum world is almost as strong as it is in a classical world for secret-key agreement, and may be even stronger since the door is still open for further improved protocols as pointed out in the open questions.

Before this work, quantum computers were a big advantage for eavesdroppers: Merkle's scheme collapsed, secure classical key agreement in the random oracle model was strongly believed to be impossible, and $\Omega(N^{3/2})$ was also conjectured to be the best possible security level even when Alice and Bob are allowed to use any quantum strategy. However, in light of our protocols, the situation has changed: eavesdroppers have essentially no longer any advantage in the quantum setting. Besides, there is still hope for more improvements, making quantum mechanics even more useful for cryptographers than adversaries. Furthermore, even when legitimate parties are classical, the eavesdropper's task has become much more difficult after having been as easy as the key agreement process.

As for quantum power, it is not uncommon to conclude that quantum computers outperform their classical counterparts on some given problem. However, this work demonstrated the useful property with respect to *function composition*. The composition theorems [56, 81] prove that the speedup provided by quantum computations on two sub-problems making up a composed one is preserved. In our case, for instance, we introduced a problem that is the composition of kSUM with the unstructured search problem pSEARCH.

7.4 Life-Style/Cultural Contribution

Yes, the following deserves a section. We want to communicate a profound message!

We introduced the term **Eave** (for a *neutral-gender* eavesdropper) instead of the traditional *Eve*, which has been used (even by myself in the past) for irrelevant or ridiculous reasons at the expense of moral values. The major problem is that people, even scientific ones, often surrender to facts or habits, which may be unfair or even wrong. Unfortunately, "Eave" is not close to the french word "espion". However, this should not be an obstacle to adopting this new term, considering the good goal.

BIBLIOGRAPHY

[1] Scott Aaronson. Quantum Lower Bound for the Collision Problem. In *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*, STOC'02, pages 635–642, 2002.

[2] Scott Aaronson and Yaoyun Shi. Quantum Lower Bounds for the Collision and the Element Distinctness Problems. *Journal of the ACM*, 51(4):595–605, 2004.

[3] Andris Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64(4):750 – 767, 2002. Earlier version in STOC'00.

[4] Andris Ambainis. Quantum Walk Algorithm for Element Distinctness. In *Proceedings of the 45th annual IEEE Symposium on Foundations of Computer Science*, FOCS'04, pages 22–31, 2004.

[5] Andris Ambainis. Quantum walks and their algorithmic applications, 2004. `arXiv:quant-ph/0403120v3`.

[6] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(3):37–46, 2005.

[7] Andris Ambainis. Polynomial degree vs. quantum query complexity. *Journal of Computer and System Sciences*, 72(2):220 – 238, 2006.

[8] Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle Puzzles Are Optimal—An O(n^2)-Query Attack on Any Key Exchange from a Random Oracle. In *Advances in Cryptology — CRYPTO'09*, pages 374–390, 2009.

[9] H. Barnum, M. Saks, and M. Szegedy. Quantum decision trees and semidefinite programming. In *Proceedings of The 18th IEEE annual Conference on Computational Complexity*, pages 179–193, 2003.

[10] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum Lower Bounds by Polynomials. *ACM Journal*, 48(4):778–797, 2001.

[11] Mihir Bellare and Philip Rogaway. *Introduction to Modern Cryptography.* Lecture Notes, 2005.

[12] Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[13] Aleksandrs Belovs. Adversary Lower Bound for Element Distinctness, 2012. `arXiv:1204.5074v1`.

[14] Aleksandrs Belovs. Learning-graph-based Quantum Algorithm for element k-distnictness, 2012. `arXiv:1205.1534v1`.

[15] Aleksandrs Belovs and Robert Špalek. Adversary lower bound for the k-sum problem, 2012. `arXiv:1206.6528v2`.

[16] Charles H. Bennett and Gilles Brassard. Quantum Cryptography: Public Key Distribution and Coin Tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, volume 175, pages 175–179, 1984.

[17] Charles H. Bennett and John Gill. Relative to a Random Oracle A, $\mathbf{P}^A \neq \mathbf{NP}^A \neq$ co-$\mathbf{NP}^A$ with Probability 1. *SIAM Journal on Computing*, 10:96–113, 1981.

[18] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41 (6):1915–1923, November 1995.

[19] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and Weaknesses of Quantum Computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[20] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight Bounds on Quantum Searching. *Fortschritte Der Physik*, 46:493–505, 1998.

[21] Gilles Brassard. A Time-luck Tradeoff in Relativized Cryptography. *Journal of Computer and System Sciences*, 22(3):280–311, 1981.

[22] Gilles Brassard and Paul Brately. *Fundamentals of Algorithmics.* Prentice Hall, 1996.

[23] Gilles Brassard and Louis Salvail. Quantum Merkle Puzzles. In *Proceedings of Second International Conference on Quantum, Nano, and Micro Technologies (ICQNM08)*, pages 76–79, 2008.

[24] Gilles Brassard and Peter Høyer. An exact quantum polynomial-time algorithm for simon's problem. In *Proceedings of the Fifth Israeli Symposium on Theory of Computing and Systems*, pages 12 –23, 1997.

[25] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum Algorithm for the Collision Problem, 1997. `arXiv:quant-ph/9705002v1`.

[26] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum Amplitude Amplification and Estimation. In *Quantum Computation and Information, AMS Contemporary Mathematics*, volume 305, pages 53–74, 2002.

[27] Gilles Brassard, Anne Broadbent, and Alain Tapp. Quantum Pseudo-Telepathy. *Foundations of Physics*, 35:1877–1907, 2005.

[28] Gilles Brassard, Louis Salvail, and Alain Tapp. Oblivious Transfer à la Merkle. *Quantum, Nano, and Micro Technologies, First International Conference on*, pages 102–108, 2009.

[29] Gilles Brassard, Peter Høyer, Kassem Kalach, Marc Kaplan, Sophie Laplante, and Louis Salvail. Merkle Puzzles in a Quantum World. In *Advances in Cryptogtology — CRYPTO'11*, pages 391–410, 2011.

[30] Anne Broadbent. Quantum Pseudo-Telepathy Games. Master's thesis, Université de Montréal, 2004.

[31] Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002.

[32] Harry Buhrman, Christoph Dürr, Mark Heiligman, Peter Høyer, Frédéric Magniez, Miklós Sántha, and Ronald de Wolf. Quantum Algorithms for Element Distinctness. *SIAM Journal of Computing*, 34(6):1324–1330, 2005.

[33] Christian Cachin and Ueli Maurer. Unconditional Security Against Memory-Bounded Adversaries. In *Advances in Cryptology — CRYPTO'97*, pages 292–306, 1997.

[34] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143 – 154, 1979.

[35] Andrew Childs and Robin Kothari. Quantum Query Complexity of Minor-Closed Graph Properties. *SIAM Journal on Computing*, 41(6):1426–1450, 2012.

[36] Andrew M. Childs and Jason M. Eisenberg. Quantum algorithms for subset finding. *Quantum Information and Computation*, 5:593–604, 2005.

[37] Ivan Damgård, Serge Fehr, Louis Salvail, and Christian Schaffner. Cryptography In the Bounded Quantum-Storage Model. In *Proceedings of the 46th annual IEEE Symposium on Foundations of Computer Science*, FOCS'05, pages 449–458, 2005.

[38] Whitfield Diffie and Martin Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644 –654, 1976.

[39] Cătălin Dohotaru and Peter Høyer. Exact quantum lower bound for Grover's problem. *Quantum Information and Computation*, 9(5):533–540, 2009.

[40] Stefan Dziembowski and Ueli Maurer. Tight Security Proofs for the Bounded-Storage Model. In *Proc. 34th ACM Symposium on Theory of Computing — STOC 2002*, pages 341–350, 2002.

[41] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28:637–647, June 1985.

[42] Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Computational Geometry*, 5(3):165 – 185, 1995.

[43] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Architectures for a quantum random access memory, 2008.

[44] Oded Goldriech. *Foundations of Cryptography.* Cambridge, 2004.

[45] D. Grigoriev. Randomized Complexity Lower Bounds. In *Proc. 30th ACM Symposium on Theory of Computing*, STOC'98, pages 219–223, 1998.

[46] Lov K. Grover. Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Phys. Rev. Lett.*, 79(2):325–328, 1997.

[47] Fang-Yu Hong, Yang Xiang, Zhi-Yan Zhu, Li zhen Jiang, and Liang neng Wu. A Robust Quantum Random Access Memory, 2012. `arXiv:1201.2250v1`.

[48] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, SFCS'89, pages 230–235, 1989.

[49] Russell Impagliazzo and Steven Rudich. Limits on the Provable Consequences of One-Way Permutations. In *Proceedings of the twenty-first annual ACM Symposium on Theory of Computing*, STOC'89, pages 44–61, 1989.

[50] Antoine Joux. *Algorithmic Cryptanalysis.* Chapman & Hall/CRC, 2009.

[51] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An Introduction to Quantum Computing*. Oxford, 2007.

[52] Joe Kilian. Founding Crytpography on Oblivious Transfer. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC'88, pages 20–31, 1988.

[53] Samuel Kutin. Quantum Lower Bound for the Collision Problem with Small Range. *Theory of Computing*, 1(1):29–36, 2005.

[54] Sophie Laplante and Frédéric Magniez. Lower bounds for randomized and quantum query complexity using Kolmogorov arguments. In *Proceedings of the 19th IEEE Annual Conference on Computational Complexity*, pages 294–304, 2004.

[55] C. Lavor, L.R.U. Manssur, and R. Portugal. Grover's Algorithm: Quantum Database Search, 2003. `arXiv:quant-ph/0301079v1`.

[56] Troy Lee, Rajat Mittal, Ben W. Reichardt, Robert Špalek, and Mario Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 52nd annual IEEE Symposium on Foundations of Computer Science*, pages 344–353, 2011.

[57] Anna Lubiw and András Rácz. A Lower Bound for the Integer Element Distinctiveness Problem. *Information and Computation*, 94(1):83–92, 1991.

[58] F. Magniez and A. Nayak. Quantum complexity of testing group commutativity. *Automata, Languages and Programming*, pages 101–101, 2005.

[59] F. Magniez, A. Nayak, J. Roland, and M. Sántha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2011.

[60] Wenbo Mao. *Modern Cryptography: Theory and Practice*. HP, 2003.

[61] Ueli Maurer. Conditionally-Perfect Secrecy and a Provably-Secure Randomized Cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[62] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[63] Ralph C. Merkle. Publishing a new idea, 1974. Including a link to his 1974 CS244 Project Proposal, `http://www.merkle.com/1974`.

[64] Ralph C. Merkle. Secure Communications over Insecure Channels. *Communications of the ACM*, 21(4):294–299, 1978.

[65] J. Misra and David Gries. Finding Repeated Elements. *Science of Computer Programming*, 2(2):143 – 152, 1982.

[66] Mikio Nakahara and Tetsuo Ohmi. *Quantum computing: From Linear Algebra To Physical Realizations*. CRC Press, 2008.

[67] Rolf Oppliger. *Contemporary Cryptography*. Artech House Publishers, 2005.

[68] Michael O. Rabin. How To Exchange Secrets with Oblivious Transfer. Cryptology ePrint Archive, Report 2005/187, 1981.

[69] Eleanor G. Rieffel and Wolfgang Polak. An Introduction to Quantum Computing for Non-Physicists. *ACM Comput. Surv.*, 32(3):300–335, 2000.

[70] Miklós Sántha. Quantum walk based search algorithms. In *Proceedings of the 5th international conference on Theory and applications of models of computation*, (TAMC'08), pages 31–46, 2008.

[71] Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. In *Proceedings of the 43rd Annual Symposium on Foundations of Computer Science*, pages 513–519, 2002.

[72] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[73] Miroslava Sotakova. Breaking One-Round Key-Agreement Protocols in the Random Oracle Model. Cryptology ePrint Archive, Report 2008/053, 2008.

[74] Douglas R. Stinson. *Cryptography: Theory and Practice.* Chapman and Hall/CRC, Third edition, 2005.

[75] Mario Szegedy. Quantum speed-up of markov chain based algorithms. In *Proceedings of 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–41, 2004.

[76] Robert Špalek and Mario Szegedy. All Quantum Adversary Methods are Equivalent. In *Proceedings of the 32nd international conference on Automata, Languages and Programming*, pages 1299–1311, 2005.

[77] Stephen Wiesner. Conjugate coding. *Written circa 1970 and belatedly published in SIGACT News*, 15(1):78–88, 1983.

[78] Peter Høyer and Robert Špalek. Lower Bounds on Quantum Query Complexity. *Bulletin of the European Association for Theoretical Computer Science*, 87:78–103, 2005.

[79] Peter Høyer, Jan Neerbek, and Yaoyun Shi. Quantum Complexities of Ordered Searching, Sorting, and Element Distinctness. *Algorithmica, Springer-Verlag*, 34(4):429–448, 2002. Also in Proceedings of the 28th International Colloquium on Automata, Languages and Programming, LNCS 2076, 346-357, 2001.

[80] Peter Høyer, Troy Lee, and Robert Špalek. Tight adversary bounds for composite functions, 2005. `arXiv:quant-ph/0509067v3`.

[81] Peter Høyer, Troy Lee, and Robert Špalek. Negative Weights Make Adversaries Stronger. In *Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, STOC'07, pages 526–535, 2007.

[82] Shengyu Zhang. On the power of Ambainis lower bounds. *Theoretical Computer Science*, 339(2-3):241 – 256, 2005.