TÉCNICO LISBOA

Quantum Cryptography applied to Electronic-Voting Protocols

Miguel Maria Rodrigues Perlico da Cruz Sabino

Thesis to obtain the Master of Science Degree in

Mathematics and Applications

Chairperson:	Prof. António Manuel Pacheco Pires
Supervisor:	Prof. Paulo Alexandre Carreira Mateus
Member of Committee:	Prof. João Filipe Quintas dos Santos Rasga

October 2016

Acknowledgements

First of all, I would like to thank my family for their entire support, in the good and bad moments, throughout my journey in IST, for being always by my side during the Bachelor and Master degrees.

Also, I want to thank to my supervisor Professor Paulo Alexandre Carreira Mateus for having accepted to direct my thesis work, for his patience and all his guidance. Additionally, I feel grateful to all the professors who helped me define my path along this journey.

My friends who kept up along with me during the past few years, and were a huge source of motivation, support, confidence and always pushed me to be better and my two cats who were essential to safeguard my calmness and keep me focused on my long term goals.. I am thankful to you.

At last, I thank God for lighten my way and mission, giving me strength and faith to carry on.

Abstract

Remote electronic voting provides the citizens an easier way to participate in decision making processes. It helps gaping the bridge between people and all kinds of voting events, ranging from the traditional elections to referendums.

Some remote electronic voting protocols have already been proposed, and actually used in real voting events, such as the 2011 and 2013 local government elections yield in Norway, (Gjøsteen, 2011)), or the binding election in the Swiss canton of Neuchâtel, see (Galindo *et al.*, 2015), in the federal referendum conducted on March 8^{th} 2015. For the latter, it was implemented in Switzerland a protocol which is an evolution of the former used in Norway.

The major breakthrough presented in the e-voting ballot casting protocol used in Neuchâtel's referendum is its Cast-as-Intended Verification Mechanism, which allows voters to verify that their vote was cast according to their intents (making use of return codes) while keeping the single vote casting property, i.e, allowing each voter to only cast one vote.

In this thesis we analyse Neuchâtel's e-voting protocol in detail, so that we can propose some improvements (mostly related to security and verifiability concerns), which we aim to achieve mainly through the introduction of Quantum Cryptography techniques. Consequently, we will expose some quantum mechanics' concepts and laws, which will ground the security analysis on the employed quantum cryptography and then proceed to the study of Neuchâtel's protocol and respective enhancement proposals.

Keywords: electronic voting; quantum cryptography; Neuchâtel's protocol; quantum key distribution; quantum bit commitment.

Contents

Acknowledgements **i**

Abstract **iii**

1 Introduction **1**

1.1 Outline . . . 2

2 Quantum Mechanics **3**

2.1 Basic Concepts . . . 3

2.1.1 Basis states . . . 4

2.1.2 Observables . . . 4

2.1.3 Geometric representation of a qubit . . . 5

2.2 Fundamental Laws and Postulates . . . 7

3 Quantum Cryptography **11**

3.1 Quantum Key Distribution . . . 11

3.1.1 Unconditional Security . . . 12

3.2 Quantum Bit Commitment . . . 13

3.2.1 Protocol Description . . . 14

3.2.2 Security Analysis . . . 18

4 Neuchâtel's Protocol **21**

4.1 Overview . . . 21

4.2 Cryptographic Background . . . 21

4.2.1 Discrete Logarithm Problem . . . 21

4.2.2 ElGamal Cryptographic System . . . 22

4.2.3 Mix-Networks . . . 24

4.2.4 RSA-FDH . . . 25

4.2.5 Non-Interactive Zero Knowledge Proofs . . . 25

4.3 Agents . . . 27

4.4 Process Description . . . 27

4.4.1 Configuration phase . . . 27

4.4.2 Registration phase . . . 28

4.4.3 Voting phase 28
4.4.4 Counting phase 31
4.5 Honesty Requirements 31

5 Improvements on Neuchâtel's Protocol **33**
5.1 Ring Signature 33
5.2 Multi-party Computation 34
5.3 Bringing in Quantum Cryptography 36
5.3.1 Configuration phase 36
5.3.2 Registration phase 37
5.3.3 Voting phase 38
5.3.4 Counting phase 40

6 Conclusion **43**

References **45**

Chapter 1

Introduction

The possibility of turning the physical voting processes into an electronic and remote operation is of great interest in democratic societies. It would provide the citizens with a much more important role in the decision-making, since it would facilitate a lot their direct participation in every kind of voting event.

As expected, the introduction of electronic-democracy brings up some new important issues that need to be considered, as mentioned in (Riera, 2002, p. 6-7) and (Puiggalí *et al.*, 2010), namely the transparency and security related to the processes while keeping its privacy at the same time. Within e-voting perspective, information becomes electronic instead of tangible, which makes the process less transparent to observers and auditors, arising questions such as whether encrypted votes actually contains the selected voting options, or even if the decryption process properly retrieves the true vote contents. Also, the vulnerability to security issues is higher in remote systems than in standard ones. Such challenges imply the need to create special measures in order to ensure transparency of the e-voting process, as well as strong security concerns to safeguard reliable implementations. The three major verification aspects are:

- *Cast-as-Intended:* must provide the voter with methods to check if the encrypted vote prepared by the voting client application actually corresponds to the selected voting options;
- *Recorded-as-Cast:* must give the voter with assurance means that the already cast vote was properly received and stored by the remote voting server;
- *Counted-as-Recorded:* must allow voters, auditors and third party observers to audit the final result of the tally, i.e, to verify that the votes received and stored by the voting server during the voting phase are in accordance with the output result.

One of the pioneers on implementing electronic voting was Norway, which ran two trials on remote voting, in the 2011 and 2013 local government elections. The cryptographic protocol used on those, see (Gjøsteen, 2011), was designed by Scytl, a Spanish electronic voting company, with the contribution of Kristian Gjøsteen. In these attempts, the most addressed security concerns were the compromised voting devices and coercion. To deal with coercion, the protocol allows electronic revoting (each new vote cancels previously submitted ballots) or voting in paper, which cancels all electronic votes; the solution to protect against compromised voting devices is based on Return Codes, as explained by Allepuz *et al.* (2012), that can be checked by the voter against the values obtained when submitting the ballot, in order to verify that the ballot was cast as intended. Basically, this protocol relies on ElGamal encryption schemes for the ballots and mix-network before decryption to break the correlation between encrypted votes and decrypted ones.

Similarly, Switzerland also introduced remote electronic voting, having the first trials been held in 2004, in the cantons of Zurich, Geneva and Neuchâtel. In spite of the Swiss positive growth on e-voting supply (nowadays already 14 cantons provide electronic voting channels), its use has been restricted (to a maximum of 10% of the electorate) due to security reasons. Thus, the Federal Council of Switzerland developed a set of requirements aiming to expand the access to e-voting to larger portions of the eligible voters, see (Chancellery, 2013). From this guideline, we highlight that an e-voting system whose protocol allows to audit the cast-as-intended property is already able to be extended to 50% of the electorate.

In our thesis, we focus our attention on the e-voting protocol implemented for the federal referendum conducted on March 8$^{\text{th}}$ 2015, that took place in the canton of Neuchâtel, for a binding election. This was an evolution of the Norwegian protocol used in 2011 and 2013 (Gjøsteen, 2011), assuring cast-as-intended verifiability (Allepuz *et al.*, 2012), with the particularity of providing the voter with a confirmation phase which allows him to verify its voting intention before the vote is officially cast.[1] Beside that, in Neuchâtel's protocol the voter privacy does not rely on the need for two server-side entities not to collude.

On a completely different subject, quantum computation and quantum information, which are the study of the processing tasks that can be accomplished using quantum mechanical systems, are also undergoing a great development, with contributions from quantum mechanics, computer science, information theory and cryptography, see (Nielsen and Chuang, 2010). The last mentioned field, cryptography, has a special relevance for our work, since we intend to take advantage of some quantum cryptography applications in order to suggest improvements on Neuchâtel's protocol.

1.1 Outline

In Chapter 2 of this thesis we introduce some basic notions on quantum mechanics, in order to proceed for the enunciation of some major quantum laws, which will be useful to support our further reasoning. After the reader is familiarized with such ideas, in Chapter 3 we explain some quantum cryptographic techniques, specifically quantum key distribution and quantum bit commitment (Loura *et al.*, 2014), each followed by an analysis of its security properties. These will be required in the final part of this dissertation. Next, in Chapter 4 we should focus our attention in the particular protocol that had been implemented and used for a binding election in the canton of Neuchâtel, in the federal referendum conducted on March 8$^{\text{th}}$ 2015. A detailed study on this protocol is provided, based on (Galindo *et al.*, 2015), with a full examination on the used cryptography methods, as well as a careful description of the process, respective agents and honesty requirements.

In the final efforts of this thesis, Chapter 5, we propose some improvements on the Neuchâtel's protocol through the use of suitable cryptographic techniques (including quantum cryptography). For example, we will take advantage of multi-party computation to increase the protocol's security and prevent vote selling and apply quantum cryptography in certain steps of the protocol (such as key sharing) aiming to accomplish the assurance of more security requisites,[2] among others.

At last, some thoughts are exposed to conclude this dissertation in Chapter 6 alongside with some directions for future work.

[1] This feature strengthens the protocol with the possibility to only allow voters to cast one vote through the electronic channel.

[2] One of the most relevant is to guarantee that it is impossible to eavesdrop shared secrets without being noticed.

Chapter 2

Quantum Mechanics

Quantum mechanics is a mathematical framework or set of rules for the construction of physical theories. With the use of quantum systems to perform computations, instead of classical physics, a new paradigm on computation arose, the so called quantum computation.

The ultimate goal of this thesis is to study how the laws of quantum mechanics can be employed to perform cryptographic tasks and to design a proposal to include such quantum cryptography into already existing electronic voting protocol. However, in order to accomplish that we must understand first some basic ideas on quantum mechanics and understand the major laws that support quantum cryptography.

2.1 Basic Concepts

Definition 2.1. Qubit
A qubit (short for quantum bit) is physical object, such as a photon or an electron, which is the basic unit for quantum computation and quantum information.

Despite being a physical object, we shall use an abstract mathematical point of view to describe it. The qubit is the counterpart in quantum computing to the binary digit bit of classical computing. Similarly to the bit, the qubit also has a state, the difference is that the state of latter can be a superposition, i.e, a linear combination of classical states. In general the state of a qubit is described by

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2.1}$$

where $\alpha, \beta \in \mathbb{C}$. The basis $\mathcal{B} = \{|0\rangle, |1\rangle\}$ is called the computational basis.

Unlikely the classical bit case where it may be examined to determine its state (0 or 1), one cannot determine a qubit's state (values of α and β). Although a qubit can exist in a continuum of states between $|0\rangle$ and $|1\rangle$, we can only extract one bit of information from the state of a qubit. Beside that, as we will see in the next subsection, measuring a qubit will generally change its state: after the measurement, the system is in the measured state. Moreover, further measurements will always yield the same value.

Measuring the state of the qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ with respect to the computational basis, we obtain either the result 0 with probability $|\alpha|^2$ or the result 1 with probability $|\beta|^2$. The previous assertion derives the following property:

Normalization Condition: the qubit's state is a unit vector in a two-dimensional complex vector space.

Indeed, the Normalization Condition follows trivially from probability theory because the outcome of the previous measurement is a random variable R such that $R \in \{0, 1\}$, so

$$\sum_{r \in R} p(R = r) = 1 \Rightarrow |\alpha|^2 + |\beta|^2 = 1. \tag{2.2}$$

Example: Consider the qubit's state $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. Measuring the bit of this qubit yields the result 0 half of the times and the result 1 the other times, since $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$.

2.1.1 Basis states

It is worthwhile to give some attention to other measurement possibilities that one may perform, according to the chosen basis states for a qubit. Consider for example the following basis $\mathcal{B}' = \{|+\rangle, |-\rangle\}$ defined as

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Any state expressed in the computational basis $\mathcal{B}$ as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ can be re-expressed using $\mathcal{B}'$ basis:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha\frac{|+\rangle + |-\rangle}{\sqrt{2}} + \beta\frac{|+\rangle - |-\rangle}{\sqrt{2}} = \frac{\alpha + \beta}{\sqrt{2}}|+\rangle + \frac{\alpha - \beta}{\sqrt{2}}|-\rangle. \tag{2.3}$$

Similarly to the behaviour of measurements in computational basis, measuring a qubit with respect to $\mathcal{B}'$ basis yields the result "+" with probability $\frac{|\alpha+\beta|^2}{2}$ and the result "-" with probability $\frac{|\alpha-\beta|^2}{2}$. Then, the post-measurements states are $|+\rangle$ and $|-\rangle$, respectively.

Generally, having a basis $\{|a\rangle, |b\rangle\}$, any state of a qubit can be expressed through that basis, i.e, as a linear combination of the basis states: $|\psi\rangle = \alpha|a\rangle + \beta|b\rangle$. Moreover, if the basis is orthonormal (so that the Normalization Condition 2.2 is fulfilled) one may perform a measurement with respect to that basis, obtaining the expected outcomes: a with probability $|\alpha|^2$ or b with probability $|\beta|^2$.

2.1.2 Observables

In quantum physics, the main goal of performing a measurement is to obtain a value to some quantum observable.

Definition 2.2. Quantum Observable
A quantum observable is a measurable quantity, expressed as an operator, such that the property of the system state can be determined by means of an operational definition.

Examples of observables in quantum mechanics are position, velocity, momentum, spin, energy, etc. An observable determines the set of possible outcomes from measuring that observable on a quantum system prepared in a given state.

Definition 2.3. Hilbert Space
In a very simplistic way, a Hilbert space is an abstract vector space possessing the structure of an inner product that allows length and angle to be measured.

While quantum states are represent by vectors in a Hilbert space V, observables are represented by Hermitian operators on V:

Definition 2.4. Hermitian Operator
An operator A whose adjoint is A (i.e, $A = A^\dagger$) is known as a self-adjoint operator, or Hermitian operator.[1]

As already mentioned, measurement operations in quantum mechanics are irreversible and their outcome is non-deterministic, in spite of obeying to some probability distribution. According to quantum theory, each variable (such as position or velocity) corresponds to an observable, being then associated with a Hermitian operator. Suppose that A is an observable related to variable x of a qubit, and that $|a_1\rangle$ and $|a_2\rangle$ are the eigenvectors of A, with respective eigenvalues a_1 and a_2.[2] Hermitian operators satisfy some relevant properties.

Lemma 2.5. *If A is a Hermitian operator the following holds:*

1. *The eigenvalues of A are all real;*
2. *The eigenvectors of A associated to different eigenvalues are orthogonal;*
3. *The eigenstates of A form a complete set of basis states for the state space of the system.*

Therefore, one may write

$$A = a_1 \cdot |a_1\rangle\langle a_1| + a_2 \cdot |a_2\rangle\langle a_2|, \tag{2.4}$$

where A is an Hermitian operator, $|a_1\rangle$ and $|a_2\rangle$ are the eigenvectors of A, and a_1 and a_2 are the associated eigenvalues, respectively.

The values a_1 and a_2 are the possible results one might obtain for the variable x, i.e, the possible outcomes of measuring the observable A on x. Moreover, since $|a_1\rangle = 1|a_1\rangle + 0|a_2\rangle$ (resp. $|a_2\rangle = 0|a_1\rangle + 1|a_2\rangle$) we can conclude that if a measurement of A is performed upon a qubit in state $|a_1\rangle$ (resp. $|a_2\rangle$) it will return the value a_1 (resp. a_2) for sure.[3] In the case of a general state $|\psi\rangle$, the returned value is random in $\{a_1,a_2\}$, pursuant the probabilities specified in the following subsection.

In addition, if the measurement yields the result a_1 (resp. a_2), the system collapses to the eigenstate $|a_1\rangle$ (resp. $|a_2\rangle$).

From this point on, unless otherwise stated or derived from the context, we will use the qubit's basis states $|0\rangle$ and $|1\rangle$.

2.1.3 Geometric representation of a qubit

In order to help us understanding the superposition of a qubit, we may think of it through a geometric representation. Since $|\alpha|^2 + |\beta|^2 = 1$, we can rewrite the qubit's state as follows:

$$\begin{aligned}
|\psi\rangle &= \alpha|0\rangle + \beta|1\rangle = \\
&= e^{i\gamma}|\alpha||0\rangle + e^{i\gamma'}|\beta||1\rangle = e^{i\gamma}\cos\frac{\theta}{2}|0\rangle + e^{i\gamma'}\sin\frac{\theta}{2}|1\rangle = \\
&= e^{i\gamma}\cos\frac{\theta}{2}|0\rangle + e^{i\gamma}e^{i\phi}\sin\frac{\theta}{2}|1\rangle = \\
&= e^{i\gamma}\big(\cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle\big)
\end{aligned} \tag{2.5}$$

where $0 \leqslant \theta \leqslant \pi, 0 \leqslant \phi < 2\pi$ and $\gamma, \gamma' \in \mathbb{R}$, such that $\gamma + \phi = \gamma'$.

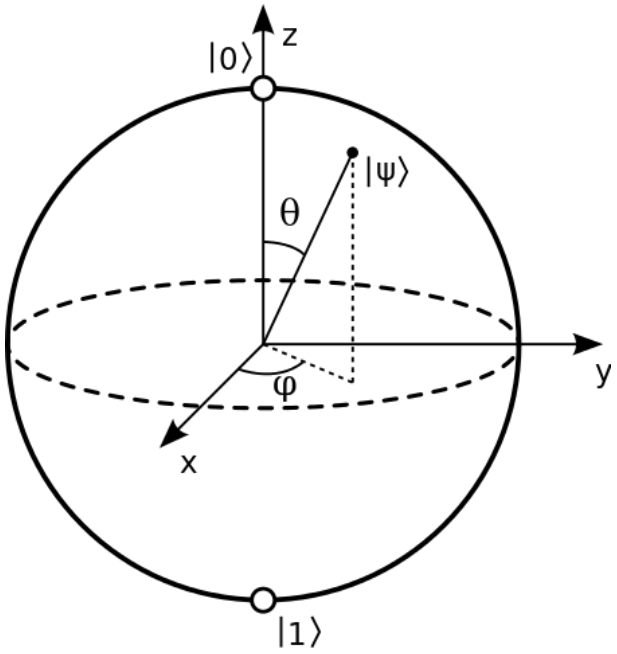
[1] It may be the case that a self-adjoint operator does not correspond to a physically meaningful observable.
[2] Note that each eigenvector trivially specifies an eigenstate.
[3] Recall that $\{|a_1\rangle, |a_2\rangle\}$ is an orthogonal basis, so the Normalization Condition holds.

Moreover, we consider that the state $e^{i\gamma}|\psi\rangle$ is the same as $|\psi\rangle$ up to the global phase factor $e^{i\gamma}$, and we shall notice that the statistics of measurement (detailed in Section 2.2) for these two states coincide. This means that the global phase factor has no observable effect, so it can actually be ignored when expressing the state. Hence, one may write

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle. \tag{2.6}$$

This is a more intuitive manner to represent the state of a single qubit, through a unit three dimensional-sphere, called the *Bloch sphere*, as illustrated in Figure 2.1, taken from Nielsen and Chuang, 2010, p.15. In this sphere the angles θ and ϕ define a point corresponding to the state, that facilitates the visualization.

Figure 2.1: Bloch sphere representation of a qubit. (Source: Nielsen and Chuang, 2010)



Some questions may arise now concerning the amount of information that a qubit represent, or even what hidden information does it conceal, i.e, the information it represents if we do not measure it (recall that measuring a qubit collapses its superposition to the measured state). These are the issues that make quantum mechanics a useful tool for computation, because when Nature evolves a closed quantum system of qubits, the states of the qubits are kept track even without extracting information through measurements, so one can get huge benefits by taking advantage of this Nature's processing of hidden information.

Now we present some notation that will be useful throughout this dissertation:

- z^*, is the complex conjugate of $z \in \mathbb{C}$;
- $|\psi\rangle$, vector (also known as a *ket*);
- $\langle\psi|$, vector dual to $|\psi\rangle$ (also known as a *bra*), $\langle\psi| = (|\psi\rangle^T)^*$;
- $\langle\phi|\psi\rangle$, inner product between the two vectors $|\phi\rangle$ and $|\psi\rangle$;
- $|\phi\rangle\langle\psi|$, outer product between the *ket* $|\phi\rangle$ and the *bra* $\langle\psi|$;
- $|\phi\rangle \otimes |\psi\rangle$, tensor product between the two vectors $|\phi\rangle$ and $|\psi\rangle$;

- $\langle\phi|A|\psi\rangle$, inner product between $|\phi\rangle$ and $A|\psi\rangle$ (equivalent to the inner product between $A^\dagger|\phi\rangle$ and $|\psi\rangle$);
- A^*, complex conjugate of the matrix A;
- A^T, transpose of the matrix A;
- $A^\dagger$, Hermitian conjugate of the matrix A, i.e, $A^\dagger = (A^T)^*$.

It is worthwhile to recall the specification of inner product within a vector space V, since it will be used many times in the future.

Definition 2.6. Inner Product
Function $(\cdot,\cdot) : V \times V \to \mathbb{C}$ *is an inner product if it satisfies:*

1. $(\cdot,\cdot)$ *is linear in the second argument, that is,* $\Big(|v\rangle, \sum_i \lambda_i |w_i\rangle\Big) = \sum_i \lambda_i (|v\rangle,|w_i\rangle)$;
2. $(|v\rangle,|w\rangle) = (|w\rangle,|v\rangle)^*$;
3. $(|v\rangle,|v\rangle) \geqslant 0$ *with equality iff* $|v\rangle = 0$.

Moreover, $\mathbb{C}$ has an inner product defined by

$$\Big((y_1,\ldots,y_n),(z_1,\ldots,z_n)\Big) \equiv \sum_i y_i^* z_i = \begin{bmatrix} y_1^* & \cdots & y_n^* \end{bmatrix} \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix}. \tag{2.7}$$

2.2 Fundamental Laws and Postulates

Now that we had already introduced some basic notions about the elementary units of quantum computation, we shall present a few laws of quantum mechanics which will be used throughout this thesis. The vast majority of the proofs are omitted, since they get out of our scope.

Such theorems and postulates will be the central key for many quantum protocols, and will be necessary for proving the security parameters of the quantum cryptography presented along this work.

Heisenberg Uncertainty Principle: *The position and momentum of a particle cannot be simultaneously measured with arbitrarily high precision. There is a minimum for the product of the uncertainties of these two measurements:*

$$\Delta x \Delta p > \frac{\hbar}{2}. \tag{2.8}$$

This uncertainty arises from the wave properties inherent in the quantum mechanical description of nature, imposing a limit to the precision with which certain pairs of physical properties (such as position and momentum) of a particle can be known at the same time, even with perfect instruments and techniques. From this principle one can derive the following theory.

Quantum Indeterminacy: *There exists a necessary incompleteness in the description of a physical system, resulting from the fact that quantum particles do not have simultaneous determinate positions and momentums.*

Quantum indeterminism is a physics' theory, verified by experience, which asserts that a certain kind of events are actually indeterministic.

Theorem 2.7. ***(No-Cloning Theorem)*** *It is impossible to make a copy of an unknown quantum state.*

Proof: Consider a quantum machine with two slots: the data slot A and the target slot B. Slot A starts with an unknown but pure quantum state $|\psi\rangle$ and slot B with a standard pure quantum state $|s\rangle$, so that the initial state of the copying machine is $|\psi\rangle \otimes |s\rangle$. The goal is to copy state $|\psi\rangle$ into slot B. Consider now the action of the unitary evolution U representing the copying procedure

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle. \tag{2.9}$$

Supposing that this copying procedure works for two particular pure states $|\psi\rangle$ and $|\phi\rangle$, one has

$$\begin{cases} U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle \\ U(|\phi\rangle \otimes |s\rangle) = |\phi\rangle \otimes |\phi\rangle \end{cases}, \tag{2.10}$$

and applying the inner product of previous equations one obtains

$$\langle\psi|\phi\rangle = (\langle\psi|\phi\rangle)^2. \tag{2.11}$$

Since the solutions to this equation are

$$x^2 = x \Leftrightarrow \begin{cases} x = 0 \\ x = 1 \end{cases}, \tag{2.12}$$

we can conclude that either $|\psi\rangle$ and $|\phi\rangle$ are orthogonal, or $|\psi\rangle = |\phi\rangle$, respectively. Thus, a general cloning device is impossible, because it would only be able to clone orthogonal states.

■

This proof essentially shows that it is impossible to clone an unknown quantum state using unitary evolution. Beside that, investigation had already led to the result that even a non-unitary cloning can only copy the states, if one admits a loss of fidelity in the cloned states.

Quantum Entanglement: *Quantum mechanical phenomenon in which the quantum states of two or more quantum particles have to be described with reference to each other, even though the individual particles may be spatially separated.*

In the case of entangled particles the quantum state of each particle cannot be described independently, but instead the whole system should be described by a quantum state. Obviously, this leads to correlations between observable physical properties of the systems.

Quantum Nonlocality: *The apparent ability of entangled quantum objects to instantaneously know about each other's state, even when separated by large distances, i.e, the ability of one particle when measured to instantly determine the state of its conjoined twin at an arbitrary distance.*

As one would expect, nonlocality occurs due to the phenomenon of entanglement, namely due to the correlations and dependencies existent between the entangled particles. This theory rejects the Principle of Locality.[4]

[4]The Principle of Locality states that distant objects cannot have direct influence on one another, meaning that an object is directly influenced only by its immediate surroundings.

Principle of Causality: *The same cause or set of causes always produces the same effects (other things being equal) and the causes temporally precedes, or is simultaneous with, its effects.*

The clause "other things being equal" refers to a variety of circumstances, such as whether the cause is sufficient or necessary, whether there is multiple causation, etc. In the field of quantum mechanics, causality is closely related to the principle of locality.

Postulate 1: *Associated to any isolated physical system is a complex vector space with inner product (that is, a Hilbert space) known as the state space of the system. The system is completely described by its state vector, which is a unit vector in the system's state space.*

Postulate 2: *The evolution of a closed quantum system is described by a unitary transformation. That is, the state $|\psi\rangle$ of the system at time t_1 is related to the state $|\psi'\rangle$ of the system at time t_2 by a unitary operator U which depends only on times t_1 and t_2, $|\psi'\rangle = U|\psi\rangle$.*

Postulate 3: *Quantum measurements are described by a collection $\{M_m\}$ of measurement operators. These are operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement then the probability that result m occurs is given by*

$$p(m) = \langle\psi|(M_m^T)^* M_m|\psi\rangle, \tag{2.13}$$

and the state of the system after the measurement is

$$\frac{M_m|\psi\rangle}{\sqrt{\langle\psi|(M_m^T)^* M_m|\psi\rangle}}. \tag{2.14}$$

The measurement operators satisfy the completeness equation

$$\sum_m (M_m^T)^* M_m = I \Longrightarrow \sum_m \langle\psi|(M_m^T)^* M_m|\psi\rangle = \sum_m p(m) = 1. \tag{2.15}$$

Postulate 4: *The state space of a composite physical system is the tensor product of the state spaces of the component physical systems. Moreover, if we have systems numbered 1 through n, and system number i is prepared in the state $|\psi_i\rangle$, then the joint state of the total system is $|\psi_1\rangle \otimes |\psi_2\rangle \otimes \ldots \otimes |\psi_n\rangle$.*

Chapter 3

Quantum Cryptography

The exploitation of the principles of quantum mechanics in order to perform cryptographic functions is the focus of quantum cryptography.

Since quantum computation allows the completion of cryptographic tasks which are conjectured to be impossible using only classical computation, it means that the majority of the best public key cryptosystems (such as *RSA* cryptosystem and *ElGamal* cryptosystem) which are based in the hardness of mathematical tasks[1], become vulnerable to quantum adversaries, who are provided with quantum algorithms running on quantum computers. Such algorithms won't be addressed in this dissertation, since it gets out of our scope.

On the other hand, quantum cryptography provides methods to accomplish provably secure protocols that allow the distribution of private information between the interacting parties, by taking advantage of the quantum mechanical principle that observation, in general, disturbs the system being observed. This is a great achievement to private key cryptography, seeing that security in private information sharing is a cornerstone in private key cryptosystems.

Next we will focus in Quantum Key Distribution, describing its most popular method followed by the analysis of its security and after that, we examine a Quantum Bit Commitment protocol, together with the assumptions and properties which ensure it to be secure.

3.1 Quantum Key Distribution

One of the major applications of quantum cryptography is quantum key distribution (QKD). While the security of classical key distribution depends on mathematical complexity (and as so, it requires some restrictions on the capabilities of an eavesdropper), the QKD can be mathematically proven to be unconditionally secure (even against quantum computers), because it relies on physical principles instead of powerful computational tasks.

Furthermore, QKD is a provably secure protocol which allows private keys to be created between two parties over a public channel, based on the fundamental idea that an eavesdropper is unable to gain any information from the intercepted qubits without disturbing their state.

The protocol described next is known as the *BB84 method* and it was proposed by Charles Bennett and Gilles Brassard in 1984.

Quantum Key Distribution, BB84: The communication involves encoding information in qubits, usually photons are used for these quantum states. In order to execute the protocol, Alice and Bob must

[1]Both cryptosystems are based on the assumption that it is difficult to invert the encryption stage if only the public key is available: *RSA* is based in the factoring problem and *ElGamal* is based on the discrete log problem.

be connected by a quantum communication channel which allows qubits to be transmitted.[2]Beside that, they are also connected by a public classical channel, such as the internet.

1. Alice and Bob agree in two pairs of quantum states, being each pair conjugate to the other pair, and the two states within a pair orthogonal to each other, such that each pair is a basis. The usual polarization state pairs used are the rectilinear basis (0 and $\frac{\pi}{2}$) and the diagonal basis ($\frac{\pi}{4}$ and $\frac{3\pi}{4}$), which are indeed conjugate to each other.

2. For the quantum transmission Alice produces random bit (0 or 1) and then randomly chooses one basis (rectilinear or diagonal) to encode it in. Depending on the bit and the basis chosen, she prepares a photon polarization state, as exemplified in Figure 3.1.

Figure 3.1: Photon polarization state according to the bit and basis selected.

Basis	1	0
Rectilinear (+)	90°	0°
Diagonal (×)	45°	135° (or -45°)

3. Alice transmits the polarized photon in the specified state to Bob, and records the state, basis and time of each photon sent. This process is repeated from the random bit stage.

4. For each received photon, Bob randomly selects a basis to measure it in (since he does not know the basis used for encoding the photons). He records the time, measurement basis and measurement result.

5. Alice and Bob communicate over the classical channel: Alice provides the basis used in the polarization of each photon, and Bob the basis in which each photon was measured in. The photon measurements (bits) where the basis didn't match are discarded by both.

6. The remaining bits constitute the shared key.

In Figure 3.2, taken from Quantum, 2009, an instantiation of the method execution is presented. As previously explained, Alice chooses random bits and random basis to encode each photon, Bob chooses random basis to measure them and finally they compare the correspondent basis, retaining only the bits where the basis coincide and using them to compose the secret key.

3.1.1 Unconditional Security

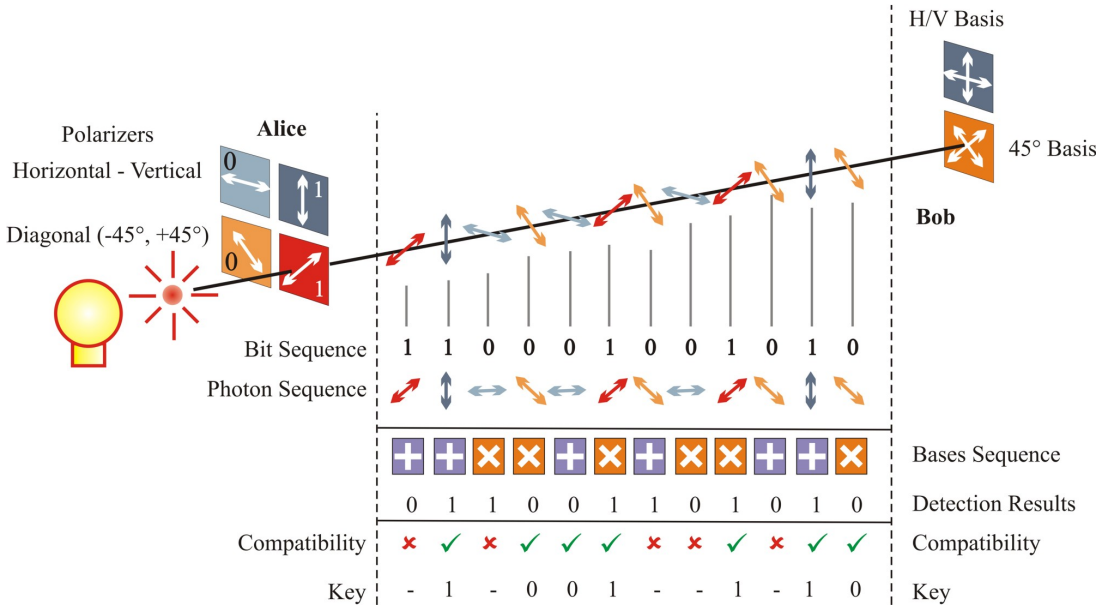
Now let's take a look on what makes this method secure, and how it allows the detection of a potential eavesdropper. First, note that the states are not all orthogonal to each other, so there is no possible measurement capable of distinguish between the four different polarization states. This is a consequence of quantum indeterminacy. To make it clear suppose that the photon measurement will be performed using the rectilinear basis: if the photon was created with rectilinear basis the measure will be the correct state (0 or $\frac{\pi}{2}$), but if it was created with diagonal basis, then the rectilinear basis will outcome 0 or $\frac{\pi}{2}$ at random.

Beside that, all the information concerning the initial polarization is lost after the measurement, due to Postulate 3.

Suppose there is an eavesdropper, Eve, trying to learn the secret key that Alice and Bob are sharing. Due to the No-Cloning Theorem (2.7), Eve cannot duplicate those photons so she must measure the intercepted ones, but according to Postulate 3, measuring an unknown quantum state will change that state.

[2]In case of photons, the channel is usually an optical fibre or just free space.

Figure 3.2: Quantum Key Distribution process. (Source: Quantum, 2009)



Consider the following scenario:

1. Alice encodes bit 0 with rectilinear basis, thus resulting in horizontal polarization (0);
2. Eve intercepts the photon and measures it, but she guesses the wrong basis, i.e, she measures this photon with diagonal basis and returns bit 1.[3] Furthermore, the photon becomes polarized in the state it was measured in, so it is now polarized in state ($\frac{\pi}{4}$).
3. Bob receives the photon and randomly chooses the rectilinear basis to measure it (with $\frac{1}{2}$ probability). The measurement yields value 1 (with $\frac{1}{2}$ probability).
4. Alice and Bob compare their basis (used to encode and measure the photon, respectively) and since they coincide, they keep the bit. But the bit Alice encoded differs from the bit Bob learned!

The previous plot shows that Alice and Bob can compare some substring of the key, searching for errors in Bob's measurements. If those errors are found they had detected the presence of an eavesdropper, so they abort the key. For more details on the security of the BB84 Quantum Key Distribution protocol see (Shor and Preskill, 2000).

3.2 Quantum Bit Commitment

In a very simplistic way, a bit commitment scheme defines a two-party method such that a party can commit to a chosen value, keeping it secret to everyone else, with the capacity to reveal its commitment later, while forcing the commitment to remain unchanged from the commitment moment until it is revealed. A lot of relevant cryptography protocols, such as Zero-Knowledge proofs, Secure Multi-Party Computation and Authentication proofs, apply bit commitment as a central part of their protocols.

In a general bit commitment protocol we can identify two general distinct phases:

- *Commitment* phase: the party who wants to do the commitment, Alice, will choose one value, usually a value of a bit (either 0 or 1). The commitment is done at a certain moment t_0.

[3] Note that this event is totally plausible, since the probability that it occurs is Pr("Eve selects wrong basis") × Pr("measure returns 1" | "basis is wrong")= $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$.

- *Opening* phase: to finalize the protocol, Alice will reveal her choice to the other party, Bob, at a later moment t_1.

Beside these phases, quantum bit commitment usually begins with an extra phase, the *Initialization* phase, which will be explained later. In order for the protocol to be secure, it must satisfy some properties: it must be *binding* and *concealing*.

- *Binding*: after Alice commits to a value, she cannot change her choice later in time, particularly during the *opening* phase.
- *Concealing*: until the *opening* phase, Bob is not able to learn anything about Alice's commitment.

It was already shown by Mayers (1997) that no quantum bit commitment protocol (QBC protocol) can be unconditionally *binding* and *concealing* at the same time, but if one considers the technological restrictions existent nowadays (and expected in the upcoming years) it is possible to design a secure QBC protocol, based on the assumptions that long-term quantum memories[4] do not exist, nor devices with the capacity to perform non-demolition measurements of photons.[5] Next, we will approach a solution for bit commitment protocols using quantum properties, proposed by Loura *et al.* (2014), whose security relies on the above mentioned technological restrictions. We will begin by describing how the protocol flows, followed by an analysis on its security.

3.2.1 Protocol Description

Recalling quantum complementarity (it is impossible to simultaneously measure two non-commuting observables[6] of a physical system), one can think of the choice on which observable to measure as the commitment to a bit value c, and even on the measurement result as a proof of the commitment. In other words, one is forced to choose only one of two possible observables, such that it will only acquire information about one of the characteristics. The previous idea has already been put to use in some cryptographic areas (for example in quantum contract signing) so that that the protocol's security relies on quantum mechanics' laws, instead of computational infeasible decryption tasks.

Before anything, some setup has to be done during the *initialization* phase, which will be specified later. Bob prepares a number of identical qubits, which will be sent to Alice in order for her to perform the measurements. These qubits should be randomly generated in one out of two quantum states, $|0\rangle$ or $|1\rangle$, with the two following requirements:

- The states $|0\rangle$ and $|1\rangle$ cannot be orthogonal.
- $\langle 0|1\rangle = \cos\theta$, with $\theta \in \left(0; \frac{\pi}{2}\right)$.

We define the states orthogonal to $|0\rangle$ and $|1\rangle$ by $|0^{\perp}\rangle$ and $|1^{\perp}\rangle$, respectively. Then, we can establish two orthogonal bases $\mathcal{B}_0 = \{|0\rangle, |0^{\perp}\rangle\}$ and $\mathcal{B}_1 = \{|1\rangle, |1^{\perp}\rangle\}$ which will produce the following two orthogonal observables:

$$\hat{C}_0 = 0 \cdot |0\rangle\langle 0| + 1 \cdot |0^{\perp}\rangle\langle 0^{\perp}|, \tag{3.1}$$

$$\hat{C}_1 = 1 \cdot |1\rangle\langle 1| + 0 \cdot |1^{\perp}\rangle\langle 1^{\perp}|. \tag{3.2}$$

[4] A quantum memory is something with the ability to store a quantum state for a given interval of time.

[5] Quantum non-demolition measurement is a special type of measurement in a quantum system in which the uncertainty of the measured observable does not increase from its measured value during the evolution of the system.

[6] Two noncommuting observables are said to be incompatible and they cannot be measured simultaneously due to the Heisenberg Uncertainty Principle.

Let $(|b_1\rangle, \ldots, |b_N\rangle)$ be the qubits sent by Bob. Alice may not receive all of them, say she only receives n ($\leq N$). Then, denoting by $(\tau_1, \ldots \tau_n)$ the Alice's measurement times and $(r_1, \ldots, r_n)$ the respective outcomes, there exists an index $k(i)$, for each $i = 1, \ldots, n$, identifying it with Bob's qubit emission time $t_{k(i)}$ and respective bit $b_{k(i)}$.

Before continuing we shall analyse the possible behaviours of the procedure, depending on the commitment and the actual state of each qubit. We can think of Alice's measurement r_i as an attempt to match the state of Bob's qubit $|b_{k(i)}\rangle$. By the previous expressions of the observables and taking into account the studied results on measurement outcomes we will use the formula presented in Postulate 3 to derive the conditional probability that measuring the observable $\hat{C}_c$ on state $|b\rangle$ one obtains result r, denoted by $p_c(r|b)$.

The eigenvectors of $\hat{C}_0$ expressed in basis $\mathcal{B}_1$ and the eigenvectors of $\hat{C}_1$ expressed in basis $\mathcal{B}_0$ are the following, respectively:

$$\begin{cases} |0\rangle = \cos\theta|1\rangle + \sin\theta|1^\perp\rangle \\ |0^\perp\rangle = e^{-i\phi}(\sin\theta|1\rangle - \cos\theta|1^\perp\rangle) \end{cases} \quad \text{and} \quad \begin{cases} |1\rangle = \cos\theta|0\rangle + e^{i\phi}\sin\theta|0^\perp\rangle \\ |1^\perp\rangle = \sin\theta|0\rangle - e^{i\phi}\cos\theta|0^\perp\rangle \end{cases}, \tag{3.3}$$

for some $\phi \in [0; 2\pi[$.

Throughout this discussion we will use the orthogonal basis $\{|0\rangle, |0^\perp\rangle\}$ whenever expressing states or constructing measurement operators. Therefore, let $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $|1\rangle = \begin{bmatrix} \cos\theta \\ e^{i\phi}\sin\theta \end{bmatrix}$.

Consider first the $\hat{C}_0$ observable. The eigenvalue associated to the eigenvector $|0\rangle$ is 0 and the eigenvalue associated to the eigenvector $|0^\perp\rangle$ is 1.

Then the measurement operators of this observable are

$$M_{0,0} = |0\rangle\langle 0| = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.4}$$

associated to output value 0, and

$$M_{0,1} = |0^\perp\rangle\langle 0^\perp| = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.5}$$

associated to output value 1.

Finally we will compute the probabilities $p_0(r|b)$:

$$\begin{aligned} p_0(0|0) = \langle 0|M_{0,0}^\dagger M_{0,0}|0\rangle = \langle 0|M_{0,0}|0\rangle = \langle 0| \left(\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) = \\ = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 1; \end{aligned} \tag{3.6}$$

$$\begin{aligned} p_0(0|1) = \langle 1|M_{0,0}^\dagger M_{0,0}|1\rangle = \langle 1|M_{0,0}|1\rangle = \langle 1| \left(\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta \\ e^{i\phi}\sin\theta \end{bmatrix} \right) = \\ = \begin{bmatrix} \cos\theta & e^{-i\phi}\sin\theta \end{bmatrix} \begin{bmatrix} \cos\theta \\ 0 \end{bmatrix} = \cos^2\theta; \end{aligned} \tag{3.7}$$

$$
\begin{aligned}
p_0(1|0) = \langle 0|M_{0,1}^{\dagger}M_{0,1}|0\rangle = \langle 0|M_{0,1}|0\rangle = \langle 0|\left(\begin{bmatrix}0 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}\right) = \\
= \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}0\\ 0\end{bmatrix} = 0;
\end{aligned}
\tag{3.8}
$$

$$
\begin{aligned}
p_0(1|1) = \langle 1|M_{0,1}^{\dagger}M_{0,1}|1\rangle = \langle 1|M_{0,1}|1\rangle = \langle 1|\left(\begin{bmatrix}0 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\cos\theta\\ e^{i\phi}\sin\theta\end{bmatrix}\right) = \\
= \begin{bmatrix}\cos\theta & e^{-i\phi}\sin\theta\end{bmatrix}\begin{bmatrix}0\\ e^{i\phi}\sin\theta\end{bmatrix} = \sin^2\theta.
\end{aligned}
\tag{3.9}
$$

Note that recalling the expressions of $|0\rangle = 1|0\rangle + 0|0^{\perp}\rangle$ and $|1\rangle = \cos\theta|0\rangle + e^{i\phi}\sin\theta|0^{\perp}\rangle$, one could use the reasoning presented in 2.1.1 to confirm the computed probabilities

$$
\begin{cases}
p_0(0|0) = p(\text{“}\,|0\rangle \text{ collapses to } |0\rangle \text{ after measuring } \hat{C}_0\text{”}) = |1|^2 = 1\\
p_0(0|1) = p(\text{“}\,|1\rangle \text{ collapses to } |0\rangle \text{ after measuring } \hat{C}_0\text{”}) = |\cos\theta|^2 = \cos^2\theta\\
p_0(1|0) = p(\text{“}\,|0\rangle \text{ collapses to } |0^{\perp}\rangle \text{ after measuring } \hat{C}_0\text{”}) = |0|^2 = 0\\
p_0(1|1) = p(\text{“}\,|1\rangle \text{ collapses to } |0^{\perp}\rangle \text{ after measuring } \hat{C}_0\text{”}) = |e^{i\phi}\sin\theta|^2 = \sin^2\theta
\end{cases}
.
\tag{3.10}
$$

Now we focus on the $\hat{C}_1$ observable. The eigenvalue associated to the eigenvector $|1\rangle$ is 1 and the eigenvalue associated to the eigenvector $|1^{\perp}\rangle$ is 0. The measurement operators of this observable are

$$
M_{1,1} = |1\rangle\langle 1| = \begin{bmatrix}\cos\theta\\ e^{i\phi}\sin\theta\end{bmatrix}\begin{bmatrix}\cos\theta & e^{-i\phi}\sin\theta\end{bmatrix} = \begin{bmatrix}\cos^2\theta & e^{-i\phi}\cos\theta\sin\theta\\ e^{i\phi}\cos\theta\sin\theta & \sin^2\theta\end{bmatrix}
\tag{3.11}
$$

associated to output value 1, and

$$
M_{1,0} = |1^{\perp}\rangle\langle 1^{\perp}| = \begin{bmatrix}\sin\theta\\ -e^{i\phi}\cos\theta\end{bmatrix}\begin{bmatrix}\sin\theta & -e^{-i\phi}\cos\theta\end{bmatrix} = \begin{bmatrix}\sin^2\theta & -e^{-i\phi}\cos\theta\sin\theta\\ -e^{i\phi}\cos\theta\sin\theta & \cos^2\theta\end{bmatrix}
\tag{3.12}
$$

associated to output value 0.

Before proceeding, it is worthwhile to compute $M_{1,1}^{\dagger}M_{1,1}$ so that we can preserve the simplicity in the subsequent probabilities calculation: the probabilities $p_1(r|b)$.

$$
\begin{aligned}
M_{1,1}^{\dagger}M_{1,1} = \begin{bmatrix}\cos^4\theta + \cos^2\theta\sin^2\theta & e^{-i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta)\\ e^{i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) & \sin^4\theta + \cos^2\theta\sin^2\theta\end{bmatrix} = \\
= \begin{bmatrix}\cos^2\theta & e^{-i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta)\\ e^{i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) & \sin^2\theta\end{bmatrix}
\end{aligned}
\tag{3.13}
$$

$$
\begin{aligned}
p_1(1|1) = \langle 1|M_{1,1}^{\dagger}M_{1,1}|1\rangle = \\
= \langle 1|\left(\begin{bmatrix}\cos^2\theta & e^{-i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta)\\ e^{i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) & \sin^2\theta\end{bmatrix}|1\rangle\right) = \\
= \langle 1|\begin{bmatrix}\cos^3\theta + \cos^3\theta\sin^2\theta + \cos\theta\sin^4\theta\\ e^{i\phi}(\cos^4\theta\sin\theta + \cos^2\theta\sin^3\theta + \sin^3\theta)\end{bmatrix} = \\
= \cos^4\theta + 2\cos^4\theta\sin^2\theta + 2\cos^2\theta\sin^4\theta + \sin^4\theta = \\
= \cos^4\theta + 2\cos^2\theta\sin^2\theta(\cos^2\theta + \sin^2\theta) + \sin^4\theta = \\
= (\cos^2\theta + \sin^2\theta)^2 = 1^2 = 1;
\end{aligned}
\tag{3.14}
$$

$$
\begin{aligned}
p_1(1|0) &= \langle 0|M_{1,1}^{\dagger}M_{1,1}|0\rangle = \\
&= \langle 0| \left(\begin{bmatrix} \cos^2\theta & e^{-i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) \\ e^{i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) & \sin^2\theta \end{bmatrix} |0\rangle \right) = \\
&= \langle 0| \begin{bmatrix} \cos^2\theta \\ e^{i\phi}(\cos^3\theta\sin\theta + \cos\theta\sin^3\theta) \end{bmatrix} = \\
&= \cos^2\theta;
\end{aligned}
\tag{3.15}
$$

For the sake of brevity, the computation of the remaining probabilities will be performed using the probability laws:

$$p_1(0|0) = 1 - p_1(1|0) = 1 - \cos^2\theta = \sin^2\theta; \tag{3.16}$$

$$p_1(0|1) = 1 - p_1(1|1) = 1 - 1 = 0. \tag{3.17}$$

Similarly to the previous case, note that $|1\rangle = 1|1\rangle + 0|1^{\perp}\rangle$ and $|0\rangle = \cos\theta|1\rangle + \sin\theta|1^{\perp}\rangle$, so using again the reasoning in 2.1.1 we can confirm that

$$
\begin{cases}
p_1(1|1) = p(\text{“}|1\rangle \text{ collapses to } |1\rangle \text{ after measuring } \hat{C}_1\text{”}) = |1|^2 = 1 \\
p_1(1|0) = p(\text{“}|0\rangle \text{ collapses to } |1\rangle \text{ after measuring } \hat{C}_1\text{”}) = |\cos\theta|^2 = \cos^2\theta \\
p_1(0|0) = p(\text{“}|0\rangle \text{ collapses to } |1^{\perp}\rangle \text{ after measuring } \hat{C}_1\text{”}) = |\sin\theta|^2 = \sin^2\theta \\
p_1(0|1) = p(\text{“}|1\rangle \text{ collapses to } |1^{\perp}\rangle \text{ after measuring } \hat{C}_1\text{”}) = |0|^2 = 0
\end{cases}
\text{ .} \tag{3.18}
$$

Analysing the obtained values, we can conclude that if the commitment c matches $b_{k(i)}$, then the measurement result r_i will be correct ($c = b_{k(i)}$), but if they do not match the result will be random according to the value of θ.

Therefore, if Alice committed to the value c, the statistics that characterize her results $\{r_i\}$ (corresponding to $\{b_{k(i)}\}$) must follow the distribution $\{p_c(r_i|b_{k(i)})\}$, so Alice's results are actually a proof of her commitment.

Now a schematic formulation of the protocol is displayed:

1. Bob generates a random string of identical qubits ($|b_1\rangle, \ldots, |b_N\rangle$), with $b_k \in \{0, 1\}$, for $k = 1, \ldots, N$, and sends them to Alice at times $(t_1, \ldots, t_N)$, respectively.

2. Alice commits to $c \in \{0, 1\}$ and according to that she measures the observable $\hat{C}_c$ on all the n qubits received. Also, she discloses the time each measurement was performed, $\tau_1 < \ldots < \tau_n$.

3. For each time measurement τ_i, with $i = 1, \ldots, n$, Bob verifies if there exists any sending time $t_{k(i)} \in \{t_1, \ldots, t_N\}$ such that $\tau_i = t_{k(i)} + vl$, where v is the speed of the qubits and l is the distance between Bob and Alice.

4. Alice reveals her commitment c and her measurement outcomes $(r_1, \ldots, r_n)$.

5. For each measurement result r_i, with $i = 1, \ldots, n$, Bob compares it with the encoded bits sent $(b_1, \ldots, b_N)$ to verify that:

$$
\begin{cases}
b_{k(i)} = c \Rightarrow r_i = c \\
b_{k(i)} \neq c \Rightarrow p_c(r_i|b_{k(i)}) \simeq \frac{n(r_i|b_{k(i)})}{n(b_{k(i)})}
\end{cases}
\text{ ,} \tag{3.19}
$$

   where $n(b_{k(i)})$ is the total number of qubits received in the state $|b_{k(i)}\rangle$ and $n(r_i|b_{k(i)})$ is the number of outcomes r_i obtained from Alice's measurements on qubits received in state $|b_{k(i)}\rangle$.

In order to understand it better, we can summarize the previous scheme in a three phased description, whose phases takes place at times T_0, T_1 and T_2, such that $T_0 < T_1 < T_2$:

- *Initialization* phase (time T_0): Bob generates a string of N random qubits $(|b_1\rangle, \ldots, |b_N\rangle)$, that represents a random string of N bits, by preparing each qubit in the pure state $|b_k\rangle$, with $b_k \in \{0, 1\}$, according to the respective bit. Bob sends them to Alice, keeping record of each state $|b_k\rangle$ and respective emission time t_k. We assume that $t_1 < \ldots < t_N$.

- *Commitment* phase (time T_1): Alice commits to value 0 or 1 by choosing the observable $\hat{C}_0$ or $\hat{C}_1$, respectively and measures the chosen observable in all the n received qubits. She records the measurement results $(r_1, \ldots, r_n)$ and the time each measurement was performed $(\tau_1, \ldots, \tau_n)$[7], disclosing the latter and keeping the former secret.

- *Opening* phase (time T_2): at time T_2 such that $t_N - t_1 << T_2 - T_1$ (for security reasons explained later), Alice reveals to Bob her commitment c and the measurement results $(r_1, \ldots, r_n)$ as a proof of her commitment, so that Bob can verify the honesty of the procedure (as described in step 5 of previous scheme).

3.2.2 Security Analysis

Now that we had already described the protocol's scheme, it is time to examine what makes it secure. We start by showing that, in practical effects, the commitment must be performed at time T_1. Indeed, the non-existence of long-term stable quantum memories[8] forces Alice to measure the qubits as soon as she receives them from Bob, meaning that Alice is obligated to choose which observable to measure at the time she receives the first qubit: $\tau_1 = T_1$. Moreover, the arrival times τ_i of each qubit can be regarded as their measurement times as well, and since $\tau_n - \tau_1 \leq t_N - t_1 << T_2 - T_1$, the commitment must be done long before the *Opening* phase (at time T_2).

On top of that, the fact that the protocol requires Alice to announce the measurement times is a reinforcement for the protocol's security. Since it is impossible to detect a photon without destroying it (changing its state) due to the current infeasibility to perform non-demolition measurements, Alice is compelled to measure the qubits on their arrival if she wants to learn their arrival times.

Consider now the *binding* property. It must be the case that Alice cannot change her commitment later in time. Focusing in our particular case and considering (without loss of generality) that she commits to value 0, she may not pass the test of committing to value 1. This property is achieved by making use of quantum mechanics' laws, namely the impossibility of a measurement capable of providing Alice with the information relative to the states of all qubits received from Bob. In other words, she cannot simultaneously get knowledge of $p_0(r_i|0)$ and $p_1(r_i|1)$ for all i's, otherwise she could always obtain the correct state of the qubit.

Obviously it is mandatory that the test cases include qubits whose states correspond to Alice's commitment and qubits whose states do not correspond, in order to make an appropriate validation statistic (as explained in step 5 of the scheme). Formally speaking, the commitment tests should include $p_0(r_i|0)$ and $p_1(r_i|1)$, but also $p_0(r_i|1)$ and $p_1(r_i|0)$.

[7] Note that $\tau_1 = T_1$, i.e, the commitment phase begins when the first measurement is performed. Moreover, the first measurement of an observable locks the bit committed.

[8] Note that this technological limitation is really important because if Alice was able to store the quantum state of the received qubits, it would be possible for her to measure them later, allowing her to postpone the commitment.

At last, we focus on the *concealing* property. One can argue that Bob could take advantage of quantum entanglement in order to extract some information about Alice's measurements, but as an effect of quantum non-locality and causality it becomes evident that even if Bob measures his portion of an entangled pair, he cannot discover Alice's local measurement, i.e, Alice's commitment, as long as they are spatially distant. Moreover, Bob cannot learn anything about Alice's results $(r_1, \ldots, r_n)$ from the measurements themselves, and therefore he is unable to obtain any knowledge from her commitment until the *opening* phase.

Chapter 4

Neuchâtel's Protocol

4.1 Overview

The remote electronic voting protocol presented in the paper (Galindo *et al.*, 2015) was implemented in the canton of Neuchâtel, Switzerland, in the March 8^{th} 2015 federal referendum, to extend e-voting to a larger percentage of the electorate.

E-voting require some verifiability conditions, established by the Federal Council of Switzerland, so that these new systems can be used by up to 50%, or even 100% of the voters. More specifically, and according to (Chancellery, 2013), the protocol must provide *cast-as-intended* verification mechanisms, in order to be used by up to 50% of the electorate, and additionally provide also *recorded-as-cast* and *counted-as-recorded* verification methods if it is to be used by up to 100% of electorate.

The Neuchâtel's protocol provides *cast-as-intended* verification, which is achieved with the use of return codes that allow voters to check if their voting options were properly recorded by the voting server (obviously keeping their privacy) and only validate them if they are correct. Also, this protocol uses single vote casting, i.e, a vote is only considered to be cast after being confirmed by the voter, which avoids the need of multi-vote casting.

The presented scheme is an evolution of the Norwegian's voting protocol (see Gjøsteen, 2011), used in Norwegian elections in 2011 and 2013. The major improvement relies on the fact that it is no longer necessary to assume that two independent server-side entities do not collude to break the voter's privacy, since here the voting device is already assigned to partially compute the return codes.

4.2 Cryptographic Background

The cryptography behind Neuchâtel's protocol is based on some important cryptographic methods, mainly the ElGamal Public-key Cryptosystem, Mix-Networks, RSA Full Domain Hash signature scheme (RSA-FDH) and Non-Interactive Zero Knowledge (NIZK) proofs.

4.2.1 Discrete Logarithm Problem

Before defining the ElGamal cryptographic system we define the **Discrete Logarithm Problem** (which is the base of ElGamal cryptosystem):

Discrete Logarithm Problem *Given a multiplicative group* $(G, \cdot)$*, an element* $\gamma \in G$ *of order* n

and an element $\alpha \in \langle\gamma\rangle$, compute the unique element a with $0 \leq a \leq n-1$ such that

$$a = log_{\gamma}\, \alpha. \tag{4.1}$$

Hardness of the Discrete Logarithm Problem: Let $(G, \cdot)$ be an abelian group and consider two elements $g, h \in G$ such that g has order n and $h \in \langle g\rangle$. Then the discrete logarithm problem over G (i.e, finding $\log_g h$) is NP and until today, no one could prove it to be in P.

4.2.2 ElGamal Cryptographic System

ElGamal Cryptosystem: *Let p be a prime such that the Discrete Logarithm problem in $(\mathbb{Z}_p^*, \cdot)$ is infeasible, and let $\alpha \in \mathbb{Z}_p^*$ be a primitive element. Let the plaintext $\mathcal{P} = \mathbb{Z}_p^*$, the ciphertext $\mathcal{C} = \mathbb{Z}_p^* \times \mathbb{Z}_p^*$ and the key space $\mathcal{K} = \{(p, \alpha, a, \beta) : \beta \equiv \alpha^a \,(mod p)\}$. The values p, α and β are the public key, and a is the private key.*

For $K = (p, \alpha, a, \beta) \in \mathcal{K}$, and for a (secret) random number $k \in \mathbb{Z}_{p-1}$, the encryption function is defined by

$$e_K(x, k) = (y_1, y_2), \tag{4.2}$$

where

$$\begin{cases} y_1 \equiv \alpha^k \,(\mathrm{mod}\, p) \\ y_2 \equiv x\beta^k \,(\mathrm{mod}\, p). \end{cases} \tag{4.3}$$

For $y_1, y_2 \in \mathbb{Z}_p^*$, the decryption function is defined by

$$d_K(y_1, y_2) = y_2 (y_1^a)^{-1} \,(\mathrm{mod}\, p). \tag{4.4}$$

Note that the ElGamal cryptographic encryption function is probabilistic since a random k is generated to proceed with the encryption.

Hereinafter, we will use `KGen`, `Enc` and `Dec` algorithms as follows:

- `KGen`: having as input a group G, with generator α of prime order q of elements in $\mathbb{Z}_p^*$, where $p = 2q + 1$ is a safe prime; it outputs an ElGamal key pair (p_k, s_k), such that $pk \in G$ and $sk \in \mathbb{Z}_q$.
- `Enc`: inputting a message $x \in G$ and the public key pk, it chooses a random $k \in \mathbb{Z}_q$ and outputs the ElGamal encryption $e_K(x, k) = (y_1, y_2)$.
- `Dec`: it receives the ElGamal encrypted pair (y_1, y_2) and the private key sk and outputs the decrypted message $d_K(y_1, y_2) = x$.

It is relevant to stress the need for the randomness of k in the ElGamal encryption `Enc`. Next, we show how to attack ElGamal in the case of knowing that k is being re-used. Suppose that the random element $k \in \mathbb{Z}_q$ is known to be re-used in the ElGamal encryptions (even without knowing k itself) and let v and v' be two unknown votes. First, note that the encrypted votes (under the same public key β) are obtained as follows:

$$\texttt{Enc}(v, \beta) = (c_1 = \alpha^k, c_2 = v\beta^k)$$

$$\texttt{Enc}(v', \beta) = (c_1' = \alpha^k, c_2' = v'\beta^k)$$

Then, the attacker can easily compute $\frac{v}{v'}$ by

$$\frac{v}{v'} = c_2 \cdot (c_2')^{-1}, \tag{4.5}$$

which allows him to determine whether $v = v'$ or not. Moreover, the attacker can simulate several specific votes $v_1', v_2', \ldots$, and compare the respective encryptions with those of the target vote v until he discovers it.

One important property that can be useful to preserve privacy at the final part of the tally is the ElGamal's homomorphic encryption property. First we define an homomorphic encryption scheme:

Definition 4.1. Homomorphic Encryption Scheme

Let $(\mathcal{P}, \mathcal{C}, \mathcal{K}, e, d)$ be an encryption scheme, where $\mathcal{P}, \mathcal{C}, \mathcal{K}$ are the plaintext, ciphertext and key spaces, respectively, and e and d are the encryption and decryption functions, such that, $(\mathcal{P}, \cdot)$ and $(\mathcal{C}, \diamond)$ are groups and the encryption algorithm e is a map from $\mathcal{P}$ to $\mathcal{C}$, i.e, $e_K : \mathcal{P} \to \mathcal{C}$, where $K \in \mathcal{K}$ is the secret key (in a secret-key cryptosystem) or the public key (in a public-key cryptosystem). The encryption scheme is said to be homomorphic if for all $a, b \in \mathcal{P}$ and $K \in \mathcal{K}$ one has

$$e_K(a) \diamond e_K(b) = e_K(a \cdot b). \tag{4.6}$$

In other words, the homomorphic property means that if we first decrypt two ciphertexts, say $d_K(y_1) = x_1$ and $d_K(y_2) = x_2$, and then operate the results in the plaintext group, $x_1 \cdot x_2$, will yield the same result as if we first operate the ciphertexts, $y_1 \diamond y_2$ and then decrypt the result, $d_K(y_1 \diamond y_2)$, i.e,

$$d_K(y_1 \diamond y_2) = d_K(y_1) \cdot d_K(y_2). \tag{4.7}$$

Furthermore, the ElGamal Cryptosystem is homomorphic over multiplication, as we can easily verify: consider two ElGamal encryptions in the ciphertext $\mathbb{Z}_p^* \times \mathbb{Z}_p^*$,

$$(y_1, y_2) = (\alpha^{k_1}, x_1 \beta^{k_1})$$

and

$$(z_1, z_2) = (\alpha^{k_2}, x_2 \beta^{k_2})$$

where x_1 and x_2 are messages in $\mathbb{Z}_p^*$, and k_1, k_2 are random values in $\mathbb{Z}_{p-1}$. Then,

$$\begin{aligned} e_K(x_1) \diamond e_K(x_2) = & \\ &= (y_1, y_2) \cdot (z_1, z_2) \\ &= (y_1 z_1, y_2 z_2) \\ &= (\alpha^{k_1} \alpha^{k_2}, x_1 \beta^{k_1} x_2 \beta^{k_2}) \\ &= (\alpha^{k_1 + k_2}, x_1 x_2 \beta^{k_1 + k_2}) \end{aligned} \tag{4.8}$$

which is indeed an encryption of $x_1 \cdot x_2$.

Although we just proved ElGamal to be homomorphic over multiplication (resp. componentwise multiplication) in $\mathbb{Z}_p^*$ (resp. $\mathbb{Z}_p^* \times \mathbb{Z}_p^*$), it is easy to convert it to an additive homomorphic cryptosystem, by encrypting α^x (where α is some generator of $\mathbb{Z}_p^*$,[1] and x is the message to be encrypted) instead of

[1] Usually the same generator used in the generation of the public key.

just encrypting the message x, see (Cramer *et al.*, 1997, p. 7). Analogously to the previous case,

$$\begin{aligned} e_K(\alpha^{x_1}) \diamond e_K(\alpha^{x_2}) = \\ &= (\alpha^{k_1}, \alpha^{x_1}\beta^{k_1}) \diamond (\alpha^{k_2}, \alpha^{x_2}\beta^{k_2}) \\ &= (\alpha^{k_1}\alpha^{k_2}, \alpha^{x_1}\beta^{k_1}\alpha^{x_2}\beta^{k_2}) \\ &= (\alpha^{k_1+k_2}, \alpha^{x_1+x_2}\beta^{k_1+k_2}) \end{aligned} \tag{4.9}$$

Furthermore, in this variant, sometimes called Exponential ElGamal, when one runs the standard decryption algorithm the outcome is $\alpha^{x_1+x_2}$, so it is necessary to solve the discrete logarithm $\log_\alpha \alpha^{x_1+x_2}$ in order to recover the desired result $x_1 + x_2$, which should not be a difficult challenge as long as the messages remain small (usually true for our voting purposes).

This property provides an alternative method (instead of using mix-nets) for anonymizing the votes before decryption. In homomorphic tally the election result is then obtained without decrypting the individual votes, but decrypting the result of operating the encrypted votes, i.e, the product of all the ciphertexts is decrypted to reveal the sum of the ballots. This approach can be efficient to deal with simple elections (when the voter only has to choose one or two options), but when there are multiple voting options to handle, which is the Neuchâtel's case, the homomorphic tally becomes cumbersome. In that scenario the most popular method relies on mix-nets, addressed in subsection 4.2.3.

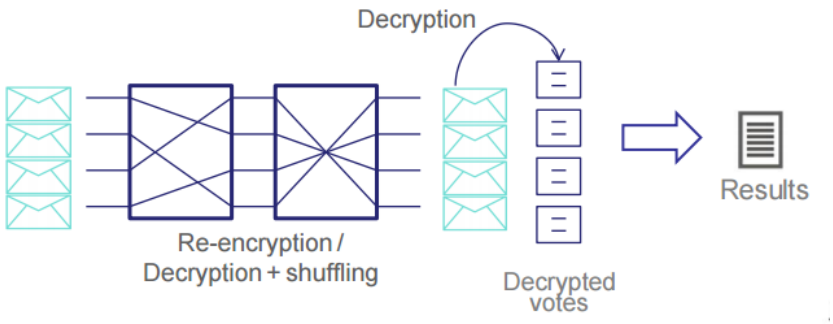
4.2.3 Mix-Networks

In this subsection we briefly describe what a mix-network is and what is its main function, although we won't further too long our discussion on this topic, since it gets out of the dissertation's scope. For more details see (D. L. Chaum, 1981)).

The purpose of a mix-network is to hide the correspondences between the items in its inputs and those in its outputs. In other words, and focusing on its specific application on e-voting, the mix-net aims to eliminate the correlation existing between the encrypted votes (given as the input) and decrypted votes (outputted by the mix-net). There are several types of mix-nets: based on nested encryption or reencryption, verifiable shuffles, probabilistic verification, etc.

In Figure 4.1 (taken from Guasch, 2011, p. 25), an illustration of a reencryption mix-net mechanism is presented.

Figure 4.1: Reencryption mix-net process. (Source: Guasch, 2011)



As displayed, the mix-net is constituted by nodes that reencrypt each received ciphertext, shuffle the new ciphertexts that resulted from the reencryption (using a secret random permutation) and then decrypt the votes. The reencyption of a vote $v = (\alpha^k, x\beta^k)$ is usually performed using ElGamal as follows

$$v' = v \diamond (\alpha^{k'}, 1 \cdot \beta^{k'}) = (\alpha^{k+k'}, x\beta^{k+k'}), \tag{4.10}$$

whose decryption obviously yields the original message x.

Note that simply scrambling the votes does not suffice, because that does not change the set of ciphertexts and so it is possible to recover (upon decryption) the votes associated to each cipher. Hence, the need to add the reencryption operation.

During this process the shuffled and reencrypted/decrypt votes output from one node are used as the input for the next node, until the votes reach the last node, where the vote contents are finally obtained decrypted.

Together with the reencryption/decryption, each node also produces a zero knowledge proof of plaintext equivalence, to demonstrate that the reencryption was performed correctly and a zero knowledge proof of correct decryption, to demonstrate the honest decryption of ciphertext. Both proofs are based on the equality of discrete logarithms, see (D. Chaum and Pedersen, 1992), and in the capacity to demonstrate in zero-knowledge the knowledge of a specific discrete logarithm, see (Schnorr, 1991). Besides that, a proof of correct shuffle is produced and together with the previous proofs, they are meant to ensure that the resulting votes output by the mix-net system actually correspond to the original votes, since they can be verified by everyone.

4.2.4 RSA-FDH

The signature scheme used in this protocol is the RSA signature algorithm with the hash variant, as known as RSA Full Domain Hash signature scheme (RSA-FDH).

RSA Signature Scheme: *Let $n = pq$, where p and q are primes. Let $\mathcal{P} = \mathcal{A} = \mathbb{Z}_n$ and define*

$$\mathcal{K} = \{(n, p, q, a, b) : n = pq,\ p \text{ and } q \text{ are primes, } ab \equiv 1\,(mod\,\phi(n))\}.$$

The values n, b are the public key and the values p, q, a are the private key. For $K = (n, p, q, a, b)$ and $x, y \in \mathbb{Z}_n$, define

$$sig_K(x) = x^a \,(\mathrm{mod}\, n) \tag{4.11}$$

and

$$ver_K(x, y) = true \leftrightarrow x \equiv y^b \,(\mathrm{mod}\, n). \tag{4.12}$$

The variant used in Neuchâtel's protocol, RSA-FDH, first hashes the message and only then signs it with the RSA Signature Scheme.

As previously, we will describe a few algorithms which will be used hereinafter:

- `SignKeyGen:` it generates a RSA key pair (pks, sks), such that $pks = \{pk_{RSA}, n_{RSA}\}$ with $n_{RSA} = pq$, where p and q are primes, and $sks = sk_{RSA}$.
- `Sign:` for a given hashing function H, it takes as input any message x and the private key sks and outputs $y = H(x)^{sks} \,(\mathrm{mod}\, n_{RSA})$.
- `SignVerify:` given the hash function H it verifies the signature, by receiving as input the public key $pks = \{pk_{RSA}, n_{RSA}\}$, the message x and the signature y and checking whether $H(x) = y^{pk_{RSA}} \,(\mathrm{mod}\, n_{RSA})$.

4.2.5 Non-Interactive Zero Knowledge Proofs

A NIZK proof is a variant of a Zero-Knowledge proof, in which no interaction is necessary between Prover and Verifier. The Prover simply sends a message to the Verifier, and the Verifier either accepts or rejects

it. A NIZK proof requires some additional structure: both the Prover and the Verifier have access to a random public string σ.

NIZK Proof System: *A NIZK proof system for input x in language L, with witness ω, is a set of polynomial algorithms (K, P, V) such that:*

1. *Key Generation: $K(1^k) \rightarrow \sigma$ generates the random public string.*
2. *Prover: $P(\sigma, x, \omega)$ produces the proof π.*
3. *Verifier: $V(\sigma, x, \pi)$ outputs $\{0, 1\}$ to accept/reject the proof.*

Which satisfy the completeness, soundness and zero-knowledge properties.

For each random string σ, define R_σ such that each element $(x, \omega) \in R_\sigma$ is a witness relation. Then:

Completeness: If the statement is true, then an honest prover can always convince an honest verifier of the statement's truthfulness. Formally:

For all k, all $\sigma \in K(1^k)$ and all $(x, \omega) \in R_\sigma$, one has $Pr\Big(P(\sigma, x, \omega) = \pi : V(\sigma, x, \pi) = 1\Big) = 1$.

Soundness: If the statement if false, no dishonest prover can convince an honest verifier of the statement's truthfulness, except with a negligible probability. Formally:

For every malicious prover $\tilde{P}$, there exists a negligible function neg[2]*, such that* $Pr\Big(K(1^k) \rightarrow \sigma,\ \tilde{P}(\sigma) \rightarrow (x, \pi) : x \notin L \wedge V(\sigma, x, \pi) = 1\Big) = neg(k)$.

Zero-Knowledge: If the statement is true, no cheating verifier can use the proof to learn anything other than the truthfulness of the statement. The formalization of this relies on the fact that every cheating verifier has a polynomial time simulator which, inputting the statement to be proven, can reproduce the interaction between the prover and the verifier, i.e, $\forall x \in L$, ω a witness for x and S a probabilistic polynomial time simulator, one has the interactions:

Between prover $\leftrightarrow$ verifier:

1. $K \rightarrow \sigma$;
2. $P(\sigma, x, \omega) \rightarrow \pi$;
3. $Output(\sigma, \pi)$.

Done by the simulator:

1. $S(1^k, x) \rightarrow (\sigma, \pi)$;
2. $Output(\sigma, \pi)$.

The result of the two interactions are thus computationally indistinguishable.

Throughout this work, the most important zero-knowledge proof systems are:

- A NIZK proof of discrete logarithm equality

$$log_{g_1} h_1 = log_{g_2} h_2 = \ldots = log_{g_n} h_n \tag{4.13}$$

  with $g_1, g_2, \ldots, g_n, h_1, h_2, \ldots, h_n$ belonging to some group G. This proof system is a generalization of the system proposed by D. Chaum and Pedersen (1992).

- A NIZK proof of knowledge of a specific discrete logarithm $log_g h$, such that g and h belong to some group G. The used system was developed by Schnorr (1991).

[2] A negligible function neg is a function $neg(k) : \mathbb{N} \rightarrow \mathbb{R}$ such that for every positive integer c there exists an integer N_c such that for all $k > N_c$, $|neg(k)| < \frac{1}{k^c}$.

4.3 Agents

The protocol has seven agents, whose roles are briefly described next:

1. Election Authorities: set up the election, compute the tally and publish the results;
2. Voters: participate in the election by choosing their voting options;
3. Registrars: provide to the voters all the information they need to vote, namely the return codes used for verifying the *cast-as-intended* integrity property;
4. Voting Server: in charge of receiving, processing and storing the ballots cast by eligible voters in the Ballot Box (BB). It may also publish some information;
5. Voting Device: casts a ballot, given the voting options chosen by the voter;
6. Code Generator: generates return codes, provided with the ballots cast by the voting device;
7. Auditors: verify the integrity of all the process.

4.4 Process Description

Neuchâtel's protocol with *cast-as-intended* verification single voting scheme is composed by four main phases: *Configuration*, *Registration*, *Voting* and *Counting* phases.

Within each phase, the agents interact between each other (through exchanges of information) and several algorithms are run. Next a step by step execution of the protocol is present.

4.4.1 Configuration phase

i Election Authorities determine the voters participating in the election, by defining the set ID of their identities.

ii Election Authorities run the $\texttt{Setup}(1^\lambda)$ algorithm: it first chooses a group G and then generates an ElGamal key pair (pk, sk), using $\texttt{KGen}(G)$; it chooses a random $x_1 \in G$ which will induce a pseudo-random function f_{x_1}, in the family F of pseudo-random functions (its purpose will become clear later) and picks two hash functions H_1 and H_2. Then, an RSA key pair is generated ($pks = \{pk_{RSA}, n_{RSA}\}, sks = sk_{RSA}$), using `SignKeyGen` and finally it defines the set $V = \{v_1, \ldots, v_k\}$ of voting options, each of them represented by a small-bit length prime in G.

In the end, the algorithm outputs the following: the election public/private key

$$(pk_e, sk_e) = ((pk, G, H_1, H_2), sk); \tag{4.14}$$

the global code generation public/private key

$$(pk_c, sk_c) = (\perp, x_1), \tag{4.15}$$

note that $f_{x_1} = f_{sk_c}$; the signing public/private key

$$(pk_s, sk_s) = (pks, sks) \tag{4.16}$$

and the set of voting options

$$V = \{v_1, \ldots, v_k\}. \tag{4.17}$$

iii Election Authorities publish pk_e, pk_c, pk_s, ID and V in the Public Bulletin Board (PBB), provide sk_c to both the Registrars and the Code Generator and also provide sk_s to the Registrar.

4.4.2 Registration phase

i Voter provides his $id \in \text{ID}$ to the Registrar, in order to participate in the election.

ii Registrars run the $\texttt{Register}(id, sk_c, sk_s)$ algorithm: it generates an ElGamal key pair (pk, sk), using $\texttt{KGen}(G)$, and chooses a random $x_2 \in G$.

The algorithm outputs: the voter's code generation public/private key

$$(pk_{id}, sk_{id}) = (pk, sk); \tag{4.18}$$

the voter's confirmation value

$$CV^{id} = x_2; \tag{4.19}$$

the voter's finalization value

$$FC^{id} = f_{sk_c}((CV^{id})^{sk_{id}}), \tag{4.20}$$

with a validity proof

$$\Pi_{FC^{id}} = \texttt{Sign}(FC^{id}, sk_s) = H_1(FC^{id})^{sk_s} \,(\text{mod}\, n_{RSA}) \tag{4.21}$$

for that finalization code; the set of voter's return codes, such that each return code is related to a voting option

$$\{v_i, RC_i^{id}\}_{i=1}^{k} = \{v_i, f_{sk_c}(v_i^{sk_{id}})\}_{i=1}^{k}, \tag{4.22}$$

and at last a set of reference values

$$\{RF_i^{id}\}_{i=1}^{k} = \{H_1(RC_i^{id})\}_{i=1}^{k}. \tag{4.23}$$

iii Registrars publish pk_{id}, $\Pi_{FC^{id}}$ and $\{RF_i^{id}\}_{i=1}^{k}$ in PBB and provide (pk_{id}, sk_{id}), $\{v_i, RC_i^{id}\}_{i=1}^{k}$, CV^{id} and FC^{id} to the voter. The set $\{\{v_i, RC_i^{id}\}_{i=1}^{k}, CV^{id}, FC^{id}\}$ constitutes the voter's Verification Card.

At this point *Configuration* and *Registration* phases are completed. Before continuing, recall that PBB has now the information

$$\left\{pk_e, pk_c, pk_s, \text{ID}, V, \left\{pk_{id}, \Pi_{FC^{id}}, \{RF_i^{id}\}_{i=1}^{k}\right\}_{id \in \text{ID}}\right\}.$$

4.4.3 Voting phase

Upon entering this phase, the user must already be prepared to authenticate himself. The authentication process is divided in two parts: the first one is handled by the citizen portal *Guichet Unique* Canton de Neuchâtel, n.d. and it provides the user with a username; the second one is handled by the electronic voting system and it involves a username/PIN-based authentication, whose PIN was obtained during the registration. We won't dwell more on this authentication subject, since it gets out of the scope of this thesis.

Assuming the user has the required information for the authentication, we shall continue the protocol's description.

i Voter authenticates through the voting device to the voting server. If the authentication is successfully completed, the values id and pk_{id} are stored in the Voting Device.

ii Voter chooses a set of voting options $\{v_{j_1}, \ldots, v_{j_t}\} \in V$ and enters them into the Voting Device, together with sk_{id}.

iii Voting Device runs the $\texttt{Vote}(id, sk_{id}, \{v_{j_1}, \ldots, v_{j_t}\})$ algorithm: it computes the vote

$$v = \prod_{l=1}^{t} v_{j_l} \tag{4.24}$$

and then encrypts it using the ElGamal encryption scheme

$$(c_1, c_2) = \texttt{Enc}(pk, v), \tag{4.25}$$

where the key pk is the one presented in the election public key pk_e. Then, a partial computation of the Return Codes[3] is calculated $(v_{j_1}^{sk_{id}}, \ldots, v_{j_t}^{sk_{id}})$ and $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is computed. Finally, three Non-Interactive Zero Knowledge (NIZK) proofs are computed:

- π_{enc} proves knowledge of the random element used for encrypting v (using the ElGamal scheme), and obtaining (c_1, c_2).
- π_{exp} proves that the ciphertext (c_1, c_2) raised to the voter's code generation private key sk_{id} is actually $(c_1^{sk_{id}}, c_2^{sk_{id}})$.
- π_{prod} proves that using the election public key pk_e to encrypt the product $\prod_{l=1}^{t} v_{j_l}^{sk_{id}}$ one obtains the ciphertext $(c_1^{sk_{id}}, c_2^{sk_{id}})$.

The π_{enc} proof is meant to assure that the encryption of the voting options was actually performed by the Voting Device. The π_{prod} proof guarantees that $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is calculated from the partial return codes corresponding to the correct voting options, and together with π_{exp} they aim to provide an evidence that the voting options contained in the ciphertext (c_1, c_2) are the same as the voting options used for the partial computation of the return codes, i.e, the voting options in the ciphertext are truly the voter's intended ones.

The algorithm outputs the ballot

$$b = \left(id, \left(c_1, c_2\right), \left(v_{j_1}^{sk_{id}}, \ldots, v_{j_t}^{sk_{id}}\right), \left(c_1^{sk_{id}}, c_2^{sk_{id}}\right), pk_{id}, \pi_{enc}, \pi_{exp}, \pi_{prod}\right). \tag{4.26}$$

Note that `Vote` is a probabilistic algorithm, due to the randomness of ElGamal encryption.

iv Voting Device sends (id, b) to the Voting Server.

v Voting Server executes $\texttt{ProcessBallot}(BB, b, id, pk_{id})$ algorithm: first it checks is there exists already

[3]Since the voting device is responsible for a partial computation of the Return Codes, the strong assumption, required in the Norwegian voting protocol, that two independent server-side entities do not collude (to preserve vote privacy) won't be necessary.

a ballot associated to id in BB.

$$\begin{cases} \text{If yes, the algorithm outputs 0 and notifies the Voting Device} \\ \text{about the error.} \\ \\ \text{If not, the algorithm validates the proofs } \pi_{enc}, \pi_{exp}, \pi_{prod} \text{ and} \\ \text{if all are successful the algorithm outputs 1.} \\ \text{In this case the ballot box BB is updated with } (id, b), \text{ with the} \\ \text{status "ballot received" and the Code Generator is notified of} \\ \text{the new update in BB.} \end{cases}$$

vi Code Generator runs $\texttt{RCGen}(b, id, sk_c)$: it generates the final Return Codes $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$, such that

$$\overline{RC_{j_l}^{id}} = f_{sk_c}(v_{j_l}^{sk_{id}}) \tag{4.27}$$

for each $l = 1, \ldots, t$. Then it checks if $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ is a subset of $\{RF_i^{id}\}_{i=1}^{k}$.

$$\begin{cases} \text{If not, the algorithm outputs } \perp \text{ and notifies the Voting Device} \\ \text{about the error/rejection.} \\ \\ \text{If yes, the algorithm outputs the unordered set of Return Codes } \{\overline{RC_{j_l}^{id}}\}_{l=1}^{t} \\ \text{and sends them to the Voting Server.} \end{cases}$$

vii Voting Server updates the status of ballot b in the BB to "return code generated" and forwards $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ to the Voting Device.

viii Voting Device shows $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ to the Voter.

ix Voter runs $\texttt{RCVerif}\left(\{v_{j_1}, \ldots, v_{j_t}\}, \{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}, \{v_i, RC_i^{id}\}_{i=1}^{k}\right)$ algorithm, which outputs:

$$\begin{cases} 1, \text{ if } \{RC_{j_i}^{id}\}_{i=1}^{t} = \{\overline{RC_{j_l}^{id}}\}_{l=1}^{t} \text{ as sets.} \\ 0, \text{ otherwise.} \end{cases}$$

If the output of $\texttt{RCVerif}$ is 1 the Voter confirms the ballot cast by providing CV^{id} to the Voting Device.

x Voting Device executes $\texttt{Confirm}(CV^{id}, id, sk_{id})$ which computes and outputs the confirmation message

$$CM^{id} = (CV^{id})^{sk_{id}}, \tag{4.28}$$

that will be sent to the Voting Server (together with id).

xi Voting Server forwards CM^{id} to the Code Generator.

xii Code Generator runs $\texttt{FCGen}(CM^{id}, id, sk_c, \Pi_{FC^{id}})$ algorithm: it uses the $\texttt{SignVerify}(pk_s, \overline{FC^{id}}, \Pi_{FC^{id}})$

algorithm, where $\overline{FC^{id}} = f_{sk_c}(CM^{id})$, outputting:

$$\begin{cases} \text{The finalization code } \overline{FC^{id}}, \text{ in case of success.} \\ \perp, \text{ otherwise. In this case the Voter is notified.} \end{cases}$$

In case of success, the finalization code $\overline{FC^{id}}$ is sent to the Voting Server.

xiii Voting Server stores $\overline{FC^{id}}$ together with the ballot b, updates the ballot's status to "confirmed" and forwards $\overline{FC^{id}}$ to the Voting Device.

xiv Voter should check if $\overline{FC^{id}}$ matches FC^{id} (received during registration). If it matches, $\overline{FC^{id}}$ serves the Voter as a confirmation of the correct submission of his vote. If it does not, the Voter may complain to election administrators, so the vote can be cast using another channel (e.g. a polling station).

4.4.4 Counting phase

i Election Authorities run $\texttt{Tally}\big(\text{BB}, sk_e, \{\Pi_{FC^{id}}\}_{id\in\text{ID}}\big)$ algorithm: for all ballots in the Ballot Box BB having a finalization code stored together with the ballot, $\texttt{SignVerify}(pk_s, \overline{FC^{id}}, \Pi_{FC^{id}})$ is run, to select the ones which have been confirmed by the voters. The resulting set of ballots is then shuffled for privacy purposes using mix-nets, see Subsection 4.2.3, (D. L. Chaum, 1981) and (Guasch, 2011, p. 25-26), and for each ballot the decryption process $\texttt{Dec}(\{c_1, c_2\}, sk_e)$ is applied, to obtain the cleartext v. After that, v is factorized to recover from $v = v_1^{\beta_1} \dots v_k^{\beta_k}$ the factors v_i such that $\beta_i = 1$, i.e, the chosen voting options. At last, the voting options v_i are used to compute the final result r, which is, together with a proof Π of the tally correctness, the output of this algorithm.

ii Auditors execute the $\texttt{Verify}(\text{PBB}, r, \Pi)$ algorithm, which outputs:

$$\begin{cases} 1, \text{ in case of success. The result } r \text{ is announced as fair.} \\ 0, \text{ otherwise. Research is made to find the reasons of failure.} \end{cases}$$

4.5 Honesty Requirements

In this subsection, an analysis of the required trust assumptions is made, in order to understand how the *cast-as-intended* verifiability and/or the privacy of the voting process would be compromised if those trust conditions were violated:

Cast-as-intended Verifiability :

To ensure *cast-as-intended* verifiability, one must consider that, for each pair presented next, at least one of the agents is honest:

- Voting Device and Code Generator: the Voting Device can produce a fake confirmation message CM^{id}, which, in the case of a dishonest Code Generator, can be used to compute a fake finalization message $\overline{FC^{id}}$ (the Code Generator does not verify its validity). In this case, the vote would be accepted without admission of the voter. In order to avoid that the Voting Device guesses a valid CM^{id} (i.e a brute force attack), the Voting Server only accept a limited number of attempts.

  Another possibility would be that the Voting Device provides Code Generator with voter's code generation private key sk_{id} and the chosen voting options. Then, Voting Device can change

the voting options and make the partial computation of RC's according to those changes, so that Voting Server will validate the proofs π_{exp} and π_{prod}, casting a changed ballot. Yet, Code Generator is able to fully generate the RC's according to the original vote, misleading the voter to confirm the vote.

- Voting Device and Registrars: since the Registrars have all the information required to compute the confirmation message CM^{id}, they could send it to the Voting Server, even if the Voter does not confirm the ballot cast (i.e. `RCVerif` outputs 0), which is forward to the Code Generator. Then the Code Generator will return the finalization code $\overline{FC^{id}}$ to Voting Server, being after forward to Voting Device. If the latter is colluding with the Registrars, $\overline{FC^{id}}$ is not showed to the Voter, leading him to think that the vote was not confirmed, when actually the vote was confirmed without his participation.

- Voting Device and Voting Server: the Voting Device can show to the Voter RC's corresponding to voter's actual vote, but that do not correspond to the cast vote (sending a ballot with encrypted options that were not used in the partial computation of RC's). Then, if the Voting Server is not honest it can validate the false proofs π_{exp} and π_{prod}, so that the vote would be changed without detection. The Voting Device can try a brute force attack (by changing the votes and trying to guess the valid RC's), but in that case the Voter would notice several wrong tries. Moreover, it suffices that the Voting Server is honest, to guarantee that Voting Device and Code Generator cannot collude to generate false RC's.

- Registrars and Code Generator: the Registrar can provide the Code Generator with the voter's confirmation value CV^{id} and the voter's code generation private key sk_{id}. In this case the Code Generator would be able to compute the confirmation message $CM^{id} = (CV^{id})^{sk_{id}}$ and then forge the finalization code $\overline{FC^{id}} = f_{sk_c}(CM^{id})$ without permission of the voter.

Privacy :

To guarantee the protocol's privacy, the following conditions are required:

- Voting Device is not compromised: since the Voter enters his voting options in the Voting Device, if the latter is compromised then the voting options can be revealed.

- Election Authorities are honest: to violate privacy, it would suffice that the Election Authorities do not shuffle the set of ballots (confirmed by the voters) before decryption, so that a correlation between encrypted votes and decrypted ones could be established.

- Verification Card contents are only known to the voter: any party who has knowledge of the computed Return Codes $\{\overline{RC^{id}_{jl}}\}^t_{l=1}$ (which will be sent to the Voter) and knowledge of the Verification Card contents, namely $\{v_i, RC^{id}_i\}^k_{i=1}$, can easily compare them to obtain the voting options chosen.

Chapter 5

Improvements on Neuchâtel's Protocol

5.1 Ring Signature

The ring signature scheme will be used with the main purpose of providing a way for the voter to verify if the information received in the Verification Card is really authentic (i.e, if it was actually sent by the Registrar), while at the same time preventing the Voter to sell the vote. Let's analyse what a ring signature is, as well as its relevant properties.

Suppose a group of entities such that each have a pair of public/private keys

$$(PK_1, SK_1), \ldots, (PK_n, SK_n).$$

A ring signature is a type of digital signature that can be performed by any entity of the group, while maintaining the anonymity, i.e, being computationally infeasible to determine which entity produced the signature.

Moreover, given the ring of public keys $R = (PK_1, \ldots, PK_n)$ and the message m, the validity of a signature σ on m can be checked by anyone, whereas creating a valid ring signature on any message, without knowing any of the secret keys, should remain unachievable. Next a formal definition is presented:

Definition 5.1. Ring Signature
A ring signature scheme is a tripe of PPT algorithms (`Gen`, `Sign`, `Vrf`) such that:

- `Gen`(1^k)*, where k is a security parameter, outputs a public key PK and a secret key SK.*
- `Sign`$_{SK_i}(M, R)$*, outputs a signature σ on the message M with respect to the ring $R = (PK_1, \ldots, PK_n)$, assuming that:*
  1. *$(R[i], SK_i)$ is a valid key pair output by `Gen`;*
  2. *$|R| \geqslant 2$;*
  3. *each public key in the ring is distinct, without loss of generality.*
- `Vrf`$_R(M, \sigma)$*, verifies a purported signature σ on a message M with respect to the ring of pubic keys R.*

The ring signature scheme must satisfy the completeness condition, which asserts that:

For any integer k, any $\{(PK_j, SK_j)\}_{j=1}^{n}$ output by $\mathtt{Gen}(1^k)$, any $i \in [n]$, and any M, we have

$$\mathtt{Vrf}_R(M, \mathtt{Sign}_{SK_i}(M, R)) = 1, \tag{5.1}$$

where $R = (PK_1, \ldots, PK_n)$.

An implementation of this scheme can be done using for example the construction presented in (Rivest *et al.*, 2001).

Ring signatures allow great flexibility because no centralized group manager or coordination among the various users is required[1] and the rings may be formed on-the-fly and in an *ad-hoc* manner.

Our suggestion is that we take the group with two entities, Voter and Registrar, considering that $R = (PK_1, PK_2)$, where $i = 1$ (resp. $i = 2$) refers to the Voter (resp. the Registrar), without loss of generality. We introduce a ring signature on the Verification Card (call it M) that the Registrar gives to the Voter at point iii of Registration phase. In other words, the Registrar signs M before giving it to the Voter, outputting σ:

$$\mathtt{Sign}_{SK_2}(M, R) = \sigma. \tag{5.2}$$

Thus we show that:

Proposition 5.2. *A ring signature with respect to the ring R=(Voter, Registrar) upon the Verification Card allows the Voter to authenticate the received information, without enabling vote selling.*

Proof (Sketch): When the Voter receives σ, he uses the verification method provided in the ring signature scheme, to validate the Verification Card.[2] If $\mathtt{Vrf}_R(M, \sigma) = 1$ then the authentication is successful.

At the same time the Voter cannot sell the authenticated vote to another party, because it is impossible to the "buyer" to discover who did sign the Verification Card due to the properties of the ring signature. Then, the Voter is unable to prove that it was the Registrar who performed the signature, which prevents vote selling.

■

5.2 Multi-party Computation

For the following work we will need to use Multi-party computation (MPC) technique: design a protocol which allows participants to compute together the value of a function over their inputs, without revealing those inputs.

Formally speaking, given a number of parties $p_1, \ldots, p_n$ each having their own secret input $x_1, \ldots, x_n$, and a public function F, the MPC aims to create a protocol such that, by interacting only between each other, the parties can correctly compute the value of $F(x_1, \ldots, x_n)$ while keeping the secrecy of their own inputs. Furthermore, each party can learn no more than just what it can learn provided that it knows its own input and the final output, i.e, they cannot learn anything more than they would if the inputs were provided to an incorruptible, perfectly trustworthy extra party who would compute the function and give the output.

We can state some major security properties that must be achieved by MPC:

[1] Indeed, users may be unaware of each other at the time they generate their public keys.

[2] Obviously the Voter knows that the signature was performed by the Registrar, since it is the only entity capable of doing so, apart from himself.

- *Input Privacy:* the exchange of messages during the execution protocol reveals no information about the private inputs provided by each party.
- *Correctness:* it is impossible to any proper subset of colluding participants to manipulate the protocol (by sharing information or deviate from the instructions) in such a way that it produces an incorrect output.
- *Independence of Inputs:* corrupted parties must choose their inputs independently of honest parties' inputs.

Now, let's take a look back at Neuchâtel's protocol, namely at point iii of Voting phase. The ElGamal encryption of the vote v is performed by the Voting Device, meaning that the Voter can easily access to the random k used for that encryption (by corrupting the Voting Device). In this case the Voter can simulate a vote v', encrypt it with ElGamal using the same k and use those informations to prove if vote v' coincides with the original vote v. This gives the voter the possibility to construct a proof of its vote, becoming able to sell it. Obviously, the random k cannot be generated by any other party, otherwise the privacy of the vote is lost.[3]

The solution for this problem is achieved with MPC. The main idea relies on the computation of a random k'' inside the "black box" of MPC, using as inputs two random numbers k and k' provided by the Voting Device and the Registrar, respectively. The value of k'' is not revealed to any party,[4] and it is immediately used to encrypt the vote entered by the Voter. The output of the MPC should be the encrypted vote, (c_1, c_2), and the proofs π_{exp} and π_{prod}, similar to the previously presented.

- π_{prod}, proves that the values $(c_1^{sk_{id}}, c_2^{sk_{id}})$ are calculated from the intended voting options, i.e, $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is the encryption of $\prod_{l=1}^{t} v_{jl}^{sk_{id}}$ under the election public key pk_e.
- π_{enc}, proves that $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is indeed the encrypted vote (c_1, c_2) raised to the voter's code generation private key sk_{id}, i.e, that the encrypted vote (c_1, c_2) truly contains the voter's intended options (considering one has already verified the correctness of π_{prod}).

Formalizing the previous idea: perform a Multi-party computation between the parties Voting Device (p_1) and Registrar (p_2), such that p_1 has private input (k, v), where k is a random element and v is Voter's vote, while p_2 has private input the random element k'. The computation should output the proofs π_{prod}, π_{enc} and the value of the public function

$$F((k, v), k') = \texttt{ElGamal}(k \oplus k', pk, v) = \texttt{ElGamal}(k'', pk, v) = (c_1, c_2), \tag{5.3}$$

where $\oplus$ is a determined operation and $\texttt{ElGamal}(k \oplus k', pk, v)$ represents the ElGamal encryption of v using $k \oplus k'$ as the secret random element required for the encryption and pk as the public key. The operation referenced as $k \oplus k'$ can be, for example, the bitwise Exclusive Or logical operation (XOR). Given two boolean variables as input, a XOR operation outputs true iff the two inputs are different. In our case, consider the binary representation of k and k', where each 1 (resp. 0) can be seen as the boolean value true (resp. false), and then apply the XOR operation to each pair of bits.[5] The *input privacy* ensures that no agent can forge the computation of k'', which could be done by providing the black box

[3] If an agent knows the k used to encrypt a vote v, it can simulate a vote v', encrypt it with the same k and the same public key, and then find out if the vote v coincides with the vote v'. The process can be repeated until the agent discovers v.

[4] Note that the *independence of inputs* ensures that no party can choose its input in order to manipulate the value of k''.

[5] Moreover, if the binary representation of the random elements k and k' is n, then the MPC will perform n XOR operations.

with a specific input to influence the result of the bitwise XOR (assuming the knowledge of the other party's input) and removes from all parties the knowledge about the encryption randomness; also, the *independence of inputs* protects the randomness of k'', preventing any cheating party from choosing its own input to manipulate the operations' result; and since k'' is not revealed to any party, it is impossible for the agents to reverse the XOR operation, i.e, to break the input privacy. Therefore, we have the following property:

Proposition 5.3. *The implementation of a Multi-party computation between Voting Device and Registrar, as described above, makes the protocol robust against vote selling, without compromising the privacy and verifiability of the vote cast.*

Proof (Sketch): In fact, the presented MPC method takes away from the Voting Device (and consequently from the Voter) the knowledge about the randomness used in the encryption of the vote, eliminating the possibility to construct a proof of the vote cast and so, the possibility to sell the vote. At the same time, it keeps the privacy of the voting options due to the lack of knowledge about the randomness used in the encryption of the vote, and also the verifiability of the correctness of the encryption (through the NIZK proofs, π_{prod} and π_{enc}, output by the MPC).

■

5.3 Bringing in Quantum Cryptography

In Chapter 2 of this thesis, several concepts on quantum mechanics were introduced followed by some important laws and postulates with the purpose of providing the required knowledge to understand the reasoning and foundations of quantum cryptography on Chapter 3. Moreover, a detailed analysis on two important techniques, Quantum Key Distribution and Quantum Bit Commitment, was performed. It is time now to make use of that.

Informally speaking, our goal now will be to reinforce the already existing e-voting protocol through the properties of the previously mentioned quantum methods. For instance, we will replace the normal exchange of messages (namely keys) by quantum key distribution, which guarantees for example the prevention of unnoticed eavesdropping, implement quantum bit commitment wherever an agent must commit to some group element/key, in such a way that any party's cheating strategy will be detected with some non-zero probability (that can be increased as desired through the use of more qubits in the protocol's execution), and introduce multi-party computation as explained previously.

In order to make our contributions as clear as possible, we will proceed to display a detailed variant of the protocol's execution, highlighting the suggested changes:

5.3.1 Configuration phase

i Election Authorities determine the voters participating in the election, by defining the set ID of their identities.

ii Election Authorities run the $\texttt{Setup}(1^\lambda)$ algorithm: it first chooses a group G and then generates an ElGamal key pair (pk, sk), using $\texttt{KGen}(G)$; it chooses a random $x_1 \in G$ which will induce a pseudo-random function f_{x_1}, in the family F of pseudo-random functions (its purpose will become clear later) and picks two hash functions H_1 and H_2. Then, an RSA key pair is generated ($pks = \{pk_{RSA}, n_{RSA}\}, sks = sk_{RSA}$), using `SignKeyGen` and finally it defines the set $V = \{v_1, \ldots, v_k\}$ of voting options, each of them represented by a small-bit length prime in G.

The establishment of (pk, sk) and (pks, sks) (during `KGen` and `SignKeyGen` executions, respectively), the choice of the random element $x_1 \in G$ (that induce f_{x_1}), the two hash functions H_1 and H_2 and the set of voting options V should be accomplished via quantum bit commitment.

In the end, the algorithm outputs the following: the election public/private key

$$(pk_e, sk_e) = ((pk, G, H_1, H_2), sk); \tag{5.4}$$

the global code generation public/private key

$$(pk_c, sk_c) = (\perp, x_1), \tag{5.5}$$

note that $f_{x_1} = f_{sk_c}$; the signing public/private key

$$(pk_s, sk_s) = (pks, sks) \tag{5.6}$$

and the set of voting options

$$V = \{v_1, \ldots, v_k\}. \tag{5.7}$$

iii Election Authorities publish pk_e, pk_c, pk_s, ID and V in the Public Bulletin Board (PBB). **Using a quantum channel, quantum key distribution is applied to provide sk_c to both the Registrars and the Code Generator and also to provide sk_s to the Registrar.**

5.3.2 Registration phase

i Voter provides his $id \in$ ID to the Registrar, in order to participate in the election.

ii Registrars run the $\texttt{Register}(id, sk_c, sk_s)$ algorithm: it generates an ElGamal key pair (pk, sk), using $\texttt{KGen}(G)$, and chooses a random $x_2 \in G$. **Again the generation of (pk, sk) within `KGen` execution is done with quantum bit commitment, as well as the choice of $x_2 \in G$.**

The algorithm outputs: the voter's code generation public/private key

$$(pk_{id}, sk_{id}) = (pk, sk); \tag{5.8}$$

the voter's confirmation value

$$CV^{id} = x_2; \tag{5.9}$$

the voter's finalization value

$$FC^{id} = f_{sk_c}((CV^{id})^{sk_{id}}), \tag{5.10}$$

with a validity proof

$$\Pi_{FC^{id}} = \texttt{Sign}(FC^{id}, sk_s) = H_1(FC^{id})^{sk_s} \,(\mathrm{mod}\, n_{RSA}) \tag{5.11}$$

for that finalization code; the set of voter's return codes, such that each return code is related to a voting option

$$\{v_i, RC_i^{id}\}_{i=1}^k = \{v_i, f_{sk_c}(v_i^{sk_{id}})\}_{i=1}^k, \tag{5.12}$$

and at last a set of reference values

$$\{RF_i^{id}\}_{i=1}^k = \{H_1(RC_i^{id})\}_{i=1}^k. \tag{5.13}$$

iii Registrars publish pk_{id}, $\Pi_{FC^{id}}$ and $\{RF_i^{id}\}_{i=1}^k$ in PBB and **provide** (pk_{id}, sk_{id})**,** $\{v_i, RC_i^{id}\}_{i=1}^k$**,** CV^{id} **and** FC^{id} **to the Voter, via quantum key distribution**.

The set $\{\{v_i, RC_i^{id}\}_{i=1}^k, CV^{id}, FC^{id}\}$ **constitutes the voter's Verification Card, and before it is sent to the Voter, it must be signed by the Registrar using a ring signature scheme with respect to the ring** $R = (PK_1, PK_2)$ **constituted by the Voter and Registrar, respectively.** More specifically, the Registrar signs the Verification Card with its own secret key SK_2,

$$\texttt{Sign}_{SK_2}(Verification\ Card, R) = \sigma. \tag{5.14}$$

iv **When the Voter receives the signed Verification Card, it authenticates the signature, using the method**

$$\texttt{Vrf}_R(Verification\ Card, \sigma). \tag{5.15}$$

If $\texttt{Vrf}_R(Verification\ Card, \sigma) = 1$ the authentication was successful and the protocol's execution can proceed. If not, an error message is produced to report that the Verification Card is invalid.

At this point *Configuration* and *Registration* phases are completed. Before continuing, recall that PBB has now the information

$$\left\{pk_e, pk_c, pk_s, \mathrm{ID}, V, \left\{pk_{id}, \Pi_{FC^{id}}, \{RF_i^{id}\}_{i=1}^k\right\}_{id\in\mathrm{ID}}\right\}.$$

5.3.3 Voting phase

Upon entering this phase, the user must already be prepared to authenticate himself. The authentication process is divided in two parts: the first one is handled by the citizen portal *Guichet Unique* Canton de Neuchâtel, n.d. and it provides the user with a username; the second one is handled by the electronic voting system and it involves a username/PIN-based authentication, whose PIN was obtained during the registration. We won't dwell more on this authentication subject, since it gets out of the scope of this thesis.

Assuming the user has the required information for the authentication, we shall continue the protocol's description.

i Voter authenticates through the voting device to the voting server. If the authentication is successfully completed, the values id and pk_{id} are stored in the Voting Device.

ii Voter chooses a set of voting options $\{v_{j_1}, \ldots, v_{j_t}\} \in V$ and enters them into the Voting Device, together with sk_{id}.

iii Voting Device runs the $\texttt{Vote}(id, sk_{id}, \{v_{j_1}, \ldots, v_{j_t}\})$ algorithm: it computes the vote

$$v = \prod_{l=1}^{t} v_{j_l} \tag{5.16}$$

and **also produces a random element** k**, whose role will be part in the encryption of the vote** v**.**

iv **The Registrar also produces a random element** k'**.**

v **The Voting Device and the Registrar give** (k, v) **and** k' **as private inputs, respectively, for a Multi-party computation (see 5.2) that produces**

$$F((k, v), k') = \texttt{ElGamal}(k \oplus k', pk, v) = \texttt{ElGamal}(k'', pk, v) = \texttt{Enc}(pk, v) = (c_1, c_2), \tag{5.17}$$

where the randomness used in the ElGamal encryption is k'' and the key pk is the one presented in the election public key pk_e. Then, a partial computation of the Return Codes[6] is calculated $(v_{j_1}^{sk_{id}}, \ldots, v_{j_t}^{sk_{id}})$ and $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is computed. Finally, three Non-Interactive Zero Knowledge (NIZK) proofs are computed:

- π_{enc} proves knowledge of the random element used for encrypting v (using the ElGamal scheme), and obtaining (c_1, c_2).
- π_{exp} proves that the ciphertext (c_1, c_2) raised to the voter's code generation private key sk_{id} is actually $(c_1^{sk_{id}}, c_2^{sk_{id}})$.
- π_{prod} proves that using the election public key pk_e to encrypt the product $\prod_{l=1}^{t} v_{j_l}^{sk_{id}}$ one obtains the ciphertext $(c_1^{sk_{id}}, c_2^{sk_{id}})$.

The π_{enc} proof is meant to assure that the encryption of the voting options was actually performed by the Voting Device. The π_{prod} proof guarantees that $(c_1^{sk_{id}}, c_2^{sk_{id}})$ is calculated from the partial return codes corresponding to the correct voting options, and together with π_{exp} they aim to provide an evidence that the voting options contained in the ciphertext (c_1, c_2) are the same as the voting options used for the partial computation of the return codes, i.e, the voting options in the ciphertext are truly the voter's intended ones.

The algorithm outputs the ballot

$$b = \left(id, \left(c_1, c_2\right), \left(v_{j_1}^{sk_{id}}, \ldots, v_{j_t}^{sk_{id}}\right), \left(c_1^{sk_{id}}, c_2^{sk_{id}}\right), pk_{id}, \pi_{enc}, \pi_{exp}, \pi_{prod} \right). \tag{5.18}$$

Note that `Vote` is a probabilistic algorithm, due to the randomness of ElGamal encryption.

vi **Voting Device sends (id, b) to the Voting Server, through quantum key distribution.**

vii Voting Server executes `ProcessBallot`(BB, b, id, pk_{id}) algorithm: first it checks is there exists already a ballot associated to id in BB.

$$\begin{cases} \text{If yes, the algorithm outputs 0 and notifies the Voting Device about the error.} \\ \\ \text{If not, the algorithm validates the proofs } \pi_{enc}, \pi_{exp}, \pi_{prod} \text{ and if all are successful the algorithm outputs 1.} \\ \text{In this case the ballot box BB is updated with } (id, b), \text{ with the status "ballot received" and the Code Generator is notified of the new update in BB.} \end{cases}$$

viii Code Generator runs `RCGen`(b, id, sk_c): it generates the final Return Codes $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$, such that

$$\overline{RC_{j_l}^{id}} = f_{sk_c}(v_{j_l}^{sk_{id}}) \tag{5.19}$$

for each $l = 1, \ldots, t$. Then it checks if $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ is a subset of $\{RF_i^{id}\}_{i=1}^{k}$.

[6]Since the voting device is responsible for a partial computation of the Return Codes, the strong assumption, required in the Norwegian voting protocol, that two independent server-side entities do not collude (to preserve vote privacy) won't be necessary.

$$\begin{cases} \text{If not, the algorithm outputs } \perp \text{ and notifies the Voting Device} \\ \text{about the error/rejection.} \\ \\ \text{If yes, the algorithm outputs the unordered set of Return Codes } \{\overline{RC_{j_l}^{id}}\}_{l=1}^{t} \\ \text{and } \textbf{sends them to the Voting Server, applying quantum key distribution.} \end{cases}$$

ix Voting Server updates the status of ballot b in the BB to "return code generated" and **executes quantum key distribution algorithm to forward $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ to the Voting Device.**

x Voting Device shows $\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t}$ to the Voter.

xi Voter runs $\texttt{RCVerif}\Big(\{v_{j_1},\ldots,v_{j_t}\},\{\overline{RC_{j_l}^{id}}\}_{l=1}^{t},\{v_i,RC_i^{id}\}_{i=1}^{k}\Big)$ algorithm, which outputs:

$$\begin{cases} 1, \text{ if } \{RC_{j_i}^{id}\}_{i=1}^{t} = \{\overline{RC_{j_l}^{id}}\}_{l=1}^{t} \text{ as sets.} \\ 0, \text{ otherwise.} \end{cases}$$

If the output of `RCVerif` is 1 the **Voter confirms the ballot cast by providing CV^{id} to the Voting Device (using quantum key distribution).**

xii Voting Device executes $\texttt{Confirm}(CV^{id}, id, sk_{id})$ which computes and outputs the confirmation message

$$CM^{id} = (CV^{id})^{sk_{id}}, \tag{5.20}$$

that will be sent to the Voting Server (together with id), via quantum key distribution.

xiii **Voting Server forwards CM^{id} to the Code Generator making use of quantum key distribution method.**

xiv Code Generator runs $\texttt{FCGen}(CM^{id}, id, sk_c, \Pi_{FC^{id}})$ algorithm: it uses the $\texttt{SignVerify}(pk_s, \overline{FC^{id}}, \Pi_{FC^{id}})$ algorithm, where $\overline{FC^{id}} = f_{sk_c}(CM^{id})$, outputting:

$$\begin{cases} \text{The finalization code } \overline{FC^{id}}, \text{ in case of success.} \\ \perp, \text{ otherwise. In this case the Voter is notified.} \end{cases}$$

In case of success, **the finalization code $\overline{FC^{id}}$ is sent to the Voting Server via quantum key distribution.**

xv Voting Server stores $\overline{FC^{id}}$ together with the ballot b, updates the ballot's status to "confirmed" and **uses quantum key distribution to forward $\overline{FC^{id}}$ to the Voting Device.**

xvi Voter should check if $\overline{FC^{id}}$ matches FC^{id} (received during registration). If it matches, $\overline{FC^{id}}$ serves the Voter as a confirmation of the correct submission of his vote. If it does not, the Voter may complain to election administrators, so the vote can be cast using another channel (e.g. a polling station).

5.3.4 Counting phase

i Election Authorities run $\texttt{Tally}\big(\text{BB}, sk_e, \{\Pi_{FC^{id}}\}_{id\in\text{ID}}\big)$ algorithm: for all ballots in the Ballot Box BB having a finalization code stored together with the ballot, $\texttt{SignVerify}(pk_s, \overline{FC^{id}}, \Pi_{FC^{id}})$ is run,

to select the ones which have been confirmed by the voters. The resulting set of ballots is then shuffled for privacy purposes using mix-nets, see Subsection 4.2.3, (D. L. Chaum, 1981) and (Guasch, 2011, p. 25-26), and for each ballot the decryption process $\texttt{Dec}(\{c_1, c_2\}, sk_e)$ is applied, to obtain the cleartext v. After that, v is factorized to recover from $v = v_1^{\beta_1} \dots v_k^{\beta_k}$ the factors v_i such that $\beta_i = 1$, i.e, the chosen voting options. At last, the voting options v_i are used to compute the final result r, which is, together with a proof Π of the tally correctness, the output of this algorithm. **The Election Authorities should commit to the announced output via quantum bit commitment, in order to guarantee the binding of the result r.**

ii Auditors execute the $\texttt{Verify}(\text{PBB}, r, \Pi)$ algorithm, which outputs:

$$\begin{cases} 1, \text{ in case of success. The result } r \text{ is announced as fair.} \\ 0, \text{ otherwise. Research is made to find the reasons of failure.} \end{cases}$$

After the presentation of this reinforced protocol we stress one last time the benefits of the implementation of quantum cryptography methods within the e-voting protocol and in the proposed schemes:

Proposition 5.4. *With the introduction of quantum cryptography to implement Multi-party computation, and quantum key distribution and quantum bit commitment within the e-voting protocol, we attain information security in the public exchanged messages for outside eavesdropper. In other words, the protocol is perfectly secure against an eavesdropper that does not take part of the protocol, since it has to break the key generated by the quantum key distribution.*

Chapter 6

Conclusion

In the last chapter we should present some final reasoning and point directions for future researching. The main goal of this dissertation was to analyse deeply an existing electronic voting protocol with *cast-as-intended* verification mechanism, (the Neuchâtel's protocol, see Chapter 4), with all its features and weaknesses, so that some suggestions could be produced and proven to be secure, in order to strengthen the protocol's performance.

A significant amount of the security increase could be achieved with the implementation of quantum methods, as exhibited in Section 5.3, namely in terms of the exchange of secret information/keys, which became shielded to unnoticed eavesdropping, and robust with respect to setup (commitment) of values with a relevant role throughout the protocol's operations. The employed quantum algorithms are explained in Chapter 3, together with a careful examination on their respective security and privacy properties. There is also an initial study concerning some concepts and postulates of quantum mechanics, whom can be found in Chapter 2, which turn out to be fundamental to comprehend the work developed.

Another valuable effort was addressed to correct some vulnerabilities found in the protocol, such as vote selling issues, or, as its counterpart, the required no-corruption assumptions. Those frailties are exposed in Section 5.1 and 5.2, followed by the solutions to overcome those problems. Built upon Ring Signature schemes and Multi-party Computation, the presented ideas bring robustness to the protocol, against the mentioned topics.

Focusing now on the next steps to take towards the strengthening of the e-voting protocol, there are a few aspects one can approach. First, one could work on the formalization and rigorous construction of proofs on the security properties achieved by the proposed methods.

Beside that, another recommendation for future work is to consider the design and optimization of the required infrastructures for the quantum interactions between agents (for example quantum communication channels that transport qubits), aiming to construct a practical and feasible e-voting system, usable by the target remote voters. As an example, we can think of transmission of single photons through an optical fiber as a simple implementation of a quantum channel. Moreover, standard fiber optics can transmit photons up to 100 km before the occurrence of losses.

Finally, and as a long-term project, this protocol could be upgraded to provide the verification mechanisms that make it suitable for a higher percentage of the electorate.

References

Allepuz, Jordi Puiggalí, Castelló, Sandra Guasch, and Voting, Scytl Secure Electronic (2012). Cast-as-Intended Verification in Norway.

Bahrami, Abasalt (2014). Quantum Nondemolition Measurements. URL: http://www.ifsc.usp.br/~strontium/Teaching/Material2014-1%20SFI5774%20Mecanicaquantica/Seminario%20-%20Abasalt%20-%20Medida%20quantica%20de%20nao-demolicao.pdf.

Busch, Paul and Lahti, Pekka (2008). Observable (Compendium entry).

Canton de Neuchâtel, République et. Neuchâtel: Guichet Unique citizen portal. URL: https://www.guichetunique.ch/public/.

Chancellery, S.F. (2013). Explications relatives à l'ordonnance de la chancellerie fédérale sur le vote électronique (OVotE). URL: https://www.bk.admin.ch/themen/pore/evoting/07979/.

Chaum, David L (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM* **24**,2, 84–90.

Chaum, David and Pedersen, Torben Pryds (1992). Wallet databases with observers. *Annual International Cryptology Conference*. Springer, 89–105.

Courteille, Ph W (2015). Mecânica Quântica.

Cramer, Ronald, Gennaro, Rosario, and Schoenmakers, Berry (1997). A secure and optimally efficient multi-authority election scheme. *European transactions on Telecommunications* **8**,5, 481–490.

Cresser, J. D. (2009). Observables and Measurements in Quantum Mechanics. URL: http://physics.mq.edu.au/~jcresser/Phys301/Chapters/Chapter13.pdf.

Dawar, Anuj (2009). Quantum Computing. URL: https://www.cl.cam.ac.uk/teaching/0910/QuantComp/notes.pdf.

Galindo, David, Guasch, Sandra, and Puiggalí, Jordi (2015). 2015 Neuchâtel's Cast-as-Intended Verification Mechanism. *E-Voting and Identity*, 3–18. Springer.

Garg, Sanjam and Nakkiran, Preetum (2014). Non-Interactive Zero-Knowledge (NIZK) and the Hidden-Bit Model. URL: http://www.cs.berkeley.edu/~sanjamg/mc/scribe/lec10.pdf.

Gjøsteen, Kristian (2011). The norwegian internet voting protocol. *E-voting and identity*, 1–18. Springer.

Guasch, Sandra (2011). Cryptographic Protocols for Transparency and Auditability in Remote Electronic Voting Schemes. URL: https://www.scytl.com/wp-content/uploads/2013/05/Cryptographic_protocols_for_pproviding_transparency_and_auditability_in_remote_electronic_voting_schemes.pdf.

Klevgard, Paul A. (2008). *Einstein's Method: A Fresh Approach to Quantum Mechanics and Relativity*. Paul A. Klevgard.

Loura, Ricardo, Almeida, Álvaro J, André, Paulo S, Pinto, Armando N, Mateus, Paulo, and Paunković, Nikola (2014). Noise and measurement errors in a practical two-state quantum bit commitment protocol. *Physical Review A* **89**,5.

Mastin, Luke (2009). Quantum Theory and the Uncertainty Principle: Nonlocality and Entanglement. [accessed Apr-2016]. URL: http://www.physicsoftheuniverse.com/topics_quantum_nonlocality.html.

Mateus, Paulo, Sernadas, Amílcar, Souto, André, and Antunes, Luís (2012). Notes on Cryptography. URL: https://fenix.tecnico.ulisboa.pt/downloadFile/3779579254156/sebenta_EN.pdf.

Mayers, Dominic (1997). Unconditionally secure quantum bit commitment is impossible. *Physical review letters* **78**,17, 3414.

Nielsen, Michael A and Chuang, Isaac L (2010). *Quantum computation and quantum information*. Cambridge university press.

Puiggalí, Jordi, Chóliz, Jesús, and Guasch, Sandra (2010). Best Practices in Internet Voting. *Scytl Secure Electronic Voting Tuset* **20**, 1–7.

Quantum, Swiss (2009). Key Sifting. [accessed May-2016]. URL: http://swissquantum.idquantique.com/?Key-Sifting.

Riera, Andreu (2002). Comments on the Report "e-Voting Security Study" Written by the Communications-Electronics Security Group, 27 pages.

Riggs, Peter J (2009). *Quantum causality: conceptual issues in the causal theory of quantum mechanics*. Vol. 23. Springer Science & Business Media.

Rivest, Ronald L, Shamir, Adi, and Tauman, Yael (2001). How to leak a secret. *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 552–565.

Schnorr, Claus-Peter (1991). Efficient signature generation by smart cards. *Journal of cryptology* **4**,3, 161–174.

Scytl. (2011). Internet Voting System with Cast as Intended Verification. URL: https://www.scytl.com/wp-content/uploads/2013/05/Internet_Voting_System_with_Cast_as_Intended_Verification_VOTEID2011.pdf.

Scytl (2015). Scytl, 2015 Neuchâtel's Cast-as-Intended Verification Mechanism. URL: http://s3c2d6c7145299d31.jimcontent.com/download/version/1441810544/module/12314567824/name/guasch1.pdf.

Shor, Peter W and Preskill, John (2000). Simple proof of security of the BB84 quantum key distribution protocol. *Physical review letters* **85**,2, 441.

Stinson, Douglas R (2005). *Cryptography: theory and practice*. CRC press.

University of Hawai'i, Hilo. Quantum Indeterminacy. [accessed Apr-2016]. URL: `http://www.uhh.hawaii.edu/~ronald/310/Quanta.htm`.

Wikipedia (2016a). Commitment Scheme | Wikipedia — The Free Encyclopedia. [accessed May-2016]. URL: `https://en.wikipedia.org/wiki/Commitment_scheme`.

— (2016b). Quantum entanglement | Wikipedia — The Free Encyclopedia. [accessed Apr-2016]. URL: `https://en.wikipedia.org/wiki/Quantum_entanglement`.

— (2016c). Quantum Key Distribution | Wikipedia — The Free Encyclopedia. [accessed May-2016]. URL: `https://en.wikipedia.org/wiki/Quantum_key_distribution`.

— (2016d). Zero-Knowledge proof | Wikipedia — The Free Encyclopedia. [accessed May-2016]. URL: `https://en.wikipedia.org/wiki/Zero-knowledge_proof`.

Yi, Xun, Paulet, Russell, and Bertino, Elisa (2014). *Homomorphic encryption and applications*. Springer.