FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

Securing IPv6 by Quantum Key Distribution for Wide Area Networks

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medizinische Informatik

eingereicht von

Gerald Dißauer
Matrikelnummer 0425724

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Gernot Salzer
Mitwirkung: Christian Kollmitzer, AIT

Wien, 23. November 2012

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Securing IPv6 by Quantum Key Distribution for Wide Area Networks

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Medical Informatics

by

Gerald Dißauer
Registration Number 0425724

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Gernot Salzer
Assistance: Christian Kollmitzer, AIT

Vienna, 23. November 2012 ______________________________ ______________________________
(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Gerald Dißauer
Vienna, Austria

Acknowledgements

This work is dedicated to my late grandfather, Franz Dißauer (* 01.04.1931 – † 28.05.2010) who is always with me in spirit and I think he would be proud of me reading this dedication. It is also dedicated to my grandmother Maria Dißauer, who did not see me really often during the work on this thesis, I will change, I promise.

I would like to thank my thesis advisor Gernot Salzer globally, but especially for his patience, critical discussions, his technical guidance and that he gave me the opportunity to work on this topic in this thesis. He guided me through the work and showed me how to work scientifically. I want to thank my second advisor Christian Kollmitzer for his technical discussions and further remarks support and Oliver Maurhart for his technical support and explanations on the AIT software and hardware environment. For his constructive discussions on the mathematics of quantum physics, I want to acknowledge Stefan Schauer.

I am especially grateful to my parents, Theresia and Helmut Dißauer, who have given me the comfort to study and letting me know that I always have their support if needed. I particularly want to thank Christa Kristöfel: Your patience, support and time have enriched my life. I also want to thank Gertrud and Karl Schmidl for our discussions and their support during my study. This work will still be unfinished without you motivating me to finish it.

Abstract

Classic cryptographic methods are based on mathematical principles i.e. they are based on unproven computational complexity assumptions. The increasing performance of attaching equipment imposes new requirements on the secure communications of the future.

The growth of the internet in recent years makes life easier for attackers who want to capture, disrupt, insert, modify, or corrupt data. The substitution of the IPv4 standard by the newer IPv6 standard leads to new fields of interest according to authentication, information security, privacy and reliable secret key distribution. IPv6 enables a large amount of unique address identifiers to set up end to end connectivity between communication endpoints. Consequently it is possible, to adress every kind of device like cars, airplanes, cell-phones, lamps, traffic lights, surgery robots, entire medical information systems or classic personal computers for instance with its unique identifier, respectively. Such unique identifiers enable access to the corresponding device via a global network. The involvment of human lives results in an increasing sensitivity and consequently the need of secure authentication, key exchange, communication and privacy to gain scalable security.

Quantum cryptography or formally known as quantum key distribution is a method to deliver secret key material between two communication partners in a provable and information theoretically secure manner. The communication partners encode the key to deliver on quantum systems and transmit qubits on an authenticated quantum channel. Classic cryptographic methods have mathematical limits whereas quantum key distribution could solve one of the major problems in classic cryptography, the reliable secret key exchange. Quantum cryptography can be integrated in existing concepts and it is an enhancement of existing classical environments and unconditionally secure algorithms such as the one time pad. The bottlenecks of quantum key distribution are the limitations in the distance and speed. The present work will show, how it is possible to secure IPv6 by quantum key distribution and bridge larger distances over optical fibres, how privacy could be realised and how unconditional security gets involved to gain scalable security on quantum networks to provide end to end connectivity over IPv6. We investigate the latency of IPv4, IPv6 and quantum key distribution to show that quantum key distribution is a potential solution to secure IPv6.

Kurzfassung

Klassische kryptographische Methoden basieren auf mathematischen Annahmen hinsichtlich der Komplexität und der damit verbundenen mathematischen Härte ein Problem zu lösen. Das bedeutet, wenn die Annahmen der mathematischen Härte widerlegt werden können, oder der technologische Fortschritt zu einer überproportionalen Steigerung der Rechenleistung führt, dass kryptographische Methoden, welche auf den Annahmen hinsichtlich der Komplexität basieren, einer immanenten Gefahr ausgesetzt sind. Diese Methoden können komprommitiert werden. Durch die Entwicklung des Internets haben Angreifer die Möglichkeit, Infrastruktur wie etwa Rechenleistung auf Bedarf zu nützen und diese dazu einzusetzen, um gegen kryptographischen Methoden vorzugehen. Neben diesen Fortschritten stellt auch die Entwicklung von Quantencomputern eine Bedrohung von klassischen, kryptographischen Methoden dar. Ein Quantencomputer ist ein Computer, der auf den Gesetzen der Quantenmechanik basiert. Die dabei eingesetzten Quantenalgorithmen ermöglichen die Kompromittierung von klassischen kryptographischen Verfahren, da eine Vielzahl an Berechnungen gleichzeitig durchgeführt werden kann.

Das Wachstum des Internets führt dazu, dass der Bedarf an freien IPv4 Adressen nicht mehr gedeckt werden kann. Die Adressknappheit erfordert neue Lösungen um den Mangel zu beseitigen. Der designierte Nachfolger von IPv4 ist das Protokoll IPv6, welches durch eine Vergrößerung des Adressraums einen Anstieg der freien IP Adressen verspricht. Die zunehmende Anzahl an Hosts im Domain Name System resultiert in einer steigenden Verbreitung von Anwendungen über das Internet. Dabei kann aufgrund der Größe des IPv6 Adressraums derzeit jeder Host im Netzwerk eindeutig identifiziert werden. Damit ist point to point Kommunikation über große Distanzen für sicherheitsrelevante Applikationen, als auch private Anwendungszwecke möglich.

Quantenkryptographie oder auch als Quantenschlüsselverteilung bezeichnet, ist eine Methode welche auf den Gesetzen der Quantenmechanik basiert. Daraus folgt, dass Quantenkryptographie beweisbar zuverlässige Schlüsselverteilung zwischen zwei Kommunikationspartnern ermöglicht. Die Information wird mit Quantenbits, auch als qubits bezeichnet vom Sender zum Empfänger übertragen. Das bedeutet, dass Quantenkryptographie informationstheoretische Sicherheit bietet wenn der verteilte Schlüssel als One Time Pad eingesetzt wird, um die Kommunikation abzusichern. Quantenkryptographie löst das Problem der sicheren Schlüsselverteilung und kann in bestehende Infrastruktur integriert werden, um die Kommunikationssicherheit zu verbessern. Bei der Distanz und der maximalen Schlüsselgenerierungsrate handelt es sich um evidente Schwachstellen der Quantenkryptographie. Die vorliegende Arbeit zeigt, dass Quantenkryptographie beweisbar sichere Schlüsselverteilung für IPv6 zur Verfügung stellt und durch die Implementierung von Quantennetzwerken das Problem der eingeschränkten Distanzen behoben werden kann. Dabei werden die Latenzzeiten von IPv4, IPv6 und Quantenkryptographie verglichen und um die Schlüsselgenerierungsrate ergänzt um zu zeigen, dass Quantenkryptographie ein potentieller Ansatz ist, um IPv6 für Punkt zu Punkt Kommunikation abzusichern.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Nowadays, a variety of security critical tasks are handled over the internet such as electronic banking or medical data processing. Cryptographic methods are involved to protect transmitted data. An interesting field in computer science is the field of cryptography. The battle between cryptographers and cryptoanalysts goes on since centuries. Cryptoanalysts might be about to overtake cryptographers through the advent of quantum computers. A quantum algorithm performs its computations in a highly parallel fashion. It poses a threat to cryptographic methods that rely on the computational complexity of calculating certain functions. However, cryptographers can protect secrets by the use of quantum key distribution which is provably secure. This connects physics to computer science.

1.2 Problem

Security critical tasks have to be protected against attacks by state of the art cryptographic techniques. Current cryptographic methods are based on number theory and rely on unproven computational complexity assumptions where numbers are chosen to obtain keys via one way functions and if the chosen numbers are too big to re-calculate the key, the cryptographic technique is assumed to be reliable. A one way function specifies a function where the inverse is hard to compute. However, there exists an inverse i.e. a decryption function to decrypt the ciphertext and obtain the plaintext. The security of IPv6 is based on the Diffie Hellman key exchange where two communication partners calculate a shared secret over probably unsecure channels via large prime numbers. Quantum computers are able to break classical cryptographic methods by the use of quantum algorithms.

Quantum key distribution relies on the quantum laws of physics and it is a promising method to protect against quantum computers and their computational power. It performs the transmission of information between two communication partners to obtain a shared secret by measuring polarizations of light quantas. That means quantum key distribution is a method to distribute a secret key between two communication partners provable secure. However, it is limited in the distance and in speed. Hence it needs to be combined with classical solutions to bridge wide areas in a global network.

1.3 Aim of the Work

Quantum key distribution is a method to provide secret key material which can be used to secure sensitive data transmission via one time pads, but it is limited in the distance. The present work shows that the internet protocol version 6 could be improved by using methods based on quantum mechanics. Quantum networks are used to bridge larger distances between two distant nodes. The work shows a performance comparison of IPv4, IPv6 and a QKD system to provide unconditional security to IPv6 and that IPv6 and quantum key distribution complement each other. That means the security of IPv6 does not necessarily depend on computational complexity assumptions.

1.4 Methodological Approach

The state of the art literature research will be followed by setting up a reference model. A requirements analysis complements an evaluation of constraints in real QKD environments. Therefore an experiment will be provided for an improved IPv6 model via open source technology to deduce, that it is actually possible to reduce the dependency on algorithmic security methods in IPv6 for wide area network communication. Performance measurements are done and statistically processed as boxplots and probability density functions.

1.5 Structure of the Work

The present work ist structured as follows. The chapter basic principles deals with the virtualization and router environment used for the experiments. In addition it deals with algebraic structures, group theory, rings and fields and Hilbert spaces. It shows the need for new cryptographic techniques to protect transmitted data against quantum computers and quantum algorithms. The internet protocol chapter shows the concepts of IPv4 and IPv6 and the header structure, respectively. It shows the Diffie Hellman key exchange protocol to agree on a shared secret over unsecure channels. The fourth chapter explains the classical cryptography and the basic mathematics of quantum information theory. It shows the differences of bits and qubits, operators, quantum gates, the basic model of quantum key distribution, the calculation rounds and the pre- and postprocessing of the quantum key exchange. An example BB84 key exchange is given and the chapter shows what happens if a third party eavesdroppes on the quantum channel. In the next chapter there are concepts shown to bridge wide areas. Quantum networks are used to connect two distant nodes. It shows that provable security is possible in the quantum network even if there are unreliable nodes in the network. The quantum point to point protocol (Q3P) is implemented by the Austrian Institute of Technology GmbH (AIT). It provides the quantum key exchange over quantum networks unconditionally secure. In the experimental evaluations chapter the results of the taken performance measurements are shown. It shows the differences in IPv4, IPv6 and the QKD system.

CHAPTER 2

Basic Principles

It is expensive to implement quantum infrastructure and make observations on real environments. A method to avoid higher costs is to virtualize an environment and make the observations in virtual machines which provide the properties of a quantum network. This section describes the basics for the experiments on quantum key distribution in virtual environments. It shows the potential of the router hardware and its optimization for the integration of quantum key distribution into IPv6. We want to describe the basic mathematical concepts to understand the necessary mathematics on quantum mechanics to deal with the measurements, taken.

2.1 Virtualization

Flexibility is a primary reason why organizations outsource resources i.e. computational tasks are ported to virtual machines. However, computing power is expensive and is another primary reason to use virtualization techniques. In information technology (IT), virtualization is a method to use the infrastructure of a physical machine (hardware and software) and separate the virtual environment from the physical one. In particular, a distinction between a physical machine and a virtual machine is drawn. A physical machine consists of an exhaustive set of standalone hardware and software, whereas a virtual machine consists of virtual hardware and software on an instance of some host hardware with an entire operating system running on it. Hence a virtual machine denotes a set of files and runs on a physical machine. It shares the physical infrastructure with other instances of virtual machines.

Virtualization - The Concept

Virtualization denotes the implementation of an environment, by contrast to a physical one. It can be divided into desktop, hardware, software, memory and storage virtualization. Desktop virtualization is referred to the concept, of separating a virtual machine from the physical machine. Figure 2.1 shows two concepts of virtualization.

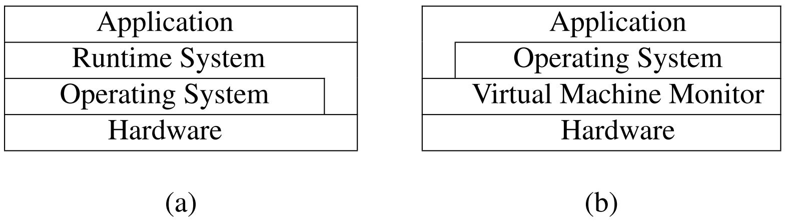


Figure 2.1: Virtualization - (a) Virtual Process Machine vs. (b) Virtual Machine Monitor

A virtual process machine uses an abstract instruction set to run instances of applications. It does this by a runtime system for a single process.

Another virtualization technique is the full virtualization via a virtual machine monitor (VMM) layer. The VMM isolates the host operating system from the instances, run in the virtualization environment. However, the VMM intercepts dedicated components and interfaces, hence it provides the complete instruction set from the host machine, network interfaces (ethx, wlanx) and other common interfaces. From a technical point of view, the benefits from virtualization are an increased portability, simplified testing and development and migration in computer science. In an economical manner, the benefits are reduced costs, a lower level of energy consumption and an increased flexibility to use the dedicated resources. By technological development, virtualization is increasingly used in a wide range of application. The concept of virtualization itself is not in the scope of this work. The interested reader can find more detailed information on virtualization in [1,45].

ORACLE Virtualbox

ORACLE virtualbox [40] is a full virtualization and cross-platform environment. It provides a virtual machine monitor, similar to VMWare player [48] and is available for Linux, Mac, OpenBSD and Windows host operating systems. As guest operating systems, virtualbox supports Microsoft Windows, Linux, Solaris, BSD, IBM OS/2, MAC OSX or DOS (Disk Operating System). Virtualbox uses harddisks for guest operating systems and they are known as `*.vdi` container, of fixed (static) or variable (dynamic) size. To enable the network adapter from a virtual machine (VM), it is necessary to specify the device the VM is attached to. The dedicated parameters are *Not attached, NAT, Bridged Adapter, Internal Network, Host-only Adapter, Generic Driver*. However, the adapters bridged adapter and internal network are of interest.

The bridged adapter uses the device driver from the host system and implements bridged networking. Because it filters the traffic it is similar to a network filter. Hence this bidirectional technique intercepts the traffic and injects the payload. A bidirectional channel specifies a point to point connection between two communication parties. Hence a bidirectional communication channel has the property that signals can be transmitted in both directions.

On the other hand, virtualbox provides an internal networking interface. It is a bidirectional networking interface, too. However, the communication is limited to a virtual network i.e. a set of virtual machines and routing devices, linked to the internal network over their interfaces. The configuration file of a virtual machine can be found in ~/VirtualBox VMs/. It is referred to as a xml file (extensible markup language) and denoted by FILENAME.vbox. The necessary parameters are the appended harddisks and the corresponding Machine uuid (universally unique identifier) to enable a unique identification on the network and the appended hard disks with the corresponding HardDisk uuid to identify the hardware. The Memory RAMSize (Random Access Memory) is specified in MB, where MB denotes Mega Byte $\left(i.e.\ 10^6 Byte\right)$. For our application 512 MB up to 1 024 MB are adequate. Page fusion avoids memory duplication, if more instances of virtual machines are running simultaneously. The last essential interface is the network adapter to specify the network parameter as a bridged interface over the physical interface eth0. To specify the used operating system (OS), the parameter OSType can be set to a list of various operating systems. This list can be found by the invocation of

```
\$ VBoxManage list ostypes.
```

Alternatives on virtualbox are VMWare [48], Microsoft Virtual PC [36] or Kernel-based Virtual Machine (KVM) [22].

2.2 OpenWRT

OpenWRT is an embedded Linux distribution for wireless routers with a command line interface (Almquist Shell, ASH) for dedicated hardware, especially developed for the Linksys WRT54GL wireless router. However, nowadays OpenWRT supports a variety of divergent hardware. For configuration there is an ASH interface. Moreover, OpenWRT is under GNU General Public Licence 2 and one can download it from https://openwrt.org/ [39] as a pre-build image, or one can build ones own for dedicated hardware support. OpenWRT provides a built-in package management (opkg) and a modyfiable file system for reconfiguration.

Architecture

After building the image, all composite configuration information are stored in a non-volatile random access memory (NVRAM). Modifications of the network interface can be made in /etc/config/network.Hence no information get lost when the power supply interrupts. Due to the versatile configuration options, OpenWRT supports the common internet protocols IPv4 and IPv6. The main IPv6 components and how OpenWRT establishes an IPv6 connection via PPP (Point to Point Protocol) is shown in Fig. 2.2. The point to point protocol (PPP) establishes a communication between two endpoints on a transmission line and is based on the IPv6 control protocol (IPV6CP). It is specified in [21]. The configuration setup information is handled by IP6CP to transmit IPv6 datagrams over the PPP link. To establish an end to end connection, the communication parties need to be uniquely identified, respectively. IPV6CP is responsible for the PPP management. It configures, enables and disables the basic IPv6 components, respectively. IPV6CP is known as a network control protocol (NCP). A link control protocol specifies a method to configure, establish and test a data-link connection between two endpoints.

Protocol	Component	Description
IPv6	PPP	Point to Point Protocol
	IP6CP	IPv6 Control Protocol
	LCP	Link Control Protocol
	NCP	Network Control Protocol

Figure 2.2: Target IPv6 Components of OpenWRT

	Parameter	Description
(X)	x86	Virtualization Environment
(X)	BCM947xx/943xx	Linksys WRT54GL Environment

Figure 2.3: Target Systems of OpenWRT

OpenWRT Configuration

By contrast to a pre-build image, a customized OpenWRT image can be build with native IPv6 support. It is recommended to build the image from the online sources with a build root environment, as a non-`root` user. After downloading the sources from the mirror [39] just invoke the setup configuration and check the missing packages as follows.

```
\$ make menuconfig
```

Finally build the image and invoke the `make` command in the terminal. More detailed build information can be found in the online documentation of OpenWRT in [39]. After creating the openWRT image for the hardware, it is necessary to clone the image for the devices to avoid conflicts. Virtualbox assigns a uuid to every machine and stores it in the setting files for each virtual machine. Two machine images with the same uuid can not run simultaneously and that is, what we need. A machine image file can not be registered to virtualbox if an image with an identical uuid already exists. The necessary packages for a native IPv6 implementation without IPv4 connectivity need to be configured in the OpenWRT configuration manager. The virtualization environment is based on a x86 architecture, whereas the physical Linksys WRT54GL router environment is based on the Broadcomm BCM947xx/943xx architecture to build the generic profile. The supported architectures are shown in a compact form in Fig. 2.3. The x86 architecture denotes a microprocessor architecture. The most common x86 central processing units (CPUs) are known from AMD and Intel. OpenWRT runs as a root-user because it does not make sense, to run a customizeable environment without root-privileges. It is based on the Intel 8086 series from the 1970ies. The specification of the target images and filesystem is necessary, to specify the instance of the target system, respectively. The grub bootloader will enable booting the environment on the dedicated machine. It is recommended to specify the parameters before the build process. Later changes will effect in a plenty of search. After building an actual OpenWRT image and installing it on VirtualBoxVM for developing and testing, boot it and check the internet connection over eth0. The most common ethernet port is denoted as eth0. It specifies a network interface and manages the ethernet network interface card (NIC). Every device has a unique configuration file. Alternatives for OpenWRT are shown in Fig. 2.4. It shows that the

Firmware	Basis	Broadcom Support
BatBox	VX Works Wind River Systems	Yes
CeroWRT	OpenWRT	Yes
DDWRT	Sveasoft Alchemy	Yes
eWRT	Sveasoft Samadhi2	Yes
FreeWRT	OpenWRT	Yes
Gargoyle	OpenWRT	Yes
HyperWRT	VX Works Wind River Systems	Yes
Midge	OpenWRT	Yes
LuCi	OpenWRT	Yes
RouterTech	MontaVista	No
Tarifa	Linksys WRT54 v1.0	Yes
Tomato	HyperWRT	Yes
Vyatta	Debian	Yes
X-WRT	OpenWRT	Yes

Figure 2.4: Firmware Alternatives to OpenWRT

most alternatives are a derivative of openWRT itself or the alternative firmware does not support the Broadcom architecture. After booting the openWRT image in a virtual machine it is recommended to check a valid connection to the internet. By pinging an arbitrary host, it can be checked if a valid internet connection exists. If there exists no valid connection, the interface options in `/etc/config/network/` need to be changed. After rebooting the system, some ping will result in an established network connection over eth0. Hence there exists a valid internet connection.

2.3 Router Environment

The WRT54GL router is a WiFi (Wireless Fidelity, also known as Wireless LAN - WLAN) device from Linksys where L means that it runs Linux on it. A router is a device which forwards data packets between a source and a destination. The source and the destination can be computer networks, too. A router is an essential device in a computer network to deliver incoming packets to the corresponding destination.

Hardware and Software

The hardware is based on a Broadcom BCM5352 central processing unit (CPU), a 16 MB (Mega Byte) random access memory and an appended 4 MB Flash memory. It has an integrated 4-port switch for internal networking and one single external networking port. The router supports ethernet and fast ethernet. Ethernet denotes set of methods to establish a network connection in a local area network. Typical transfer rates are 10 Mbit/s and in fast ethernet 100 Mbit/s (fast ethernet) per second or 1 Gbit/s (gigabit ethernet). Besides the cable oriented methods to transmit data, the linksys supports the wireless standards IEEE 802.11b and IEEE 802.11g. The 802.11

Type	Specification	Quantity
Model	WRT54GL	
	Ethernet 10/100 RJ-45	4
Ports	Internet 10/100 RJ-45	1
	Power 12VDC, 1 A	1
Cable	CAT 5	1
CPU	Broadcom 5352 200 MHz	1
RAM	16 MB	1
Flash	4 MB	1
Antenna	RP-TNC	2

Figure 2.5: Specifications of the Linksys WRT54GL

is an IEEE standard and specifies techniques for wireless data transfer. The 802.11 frequency is at 2.4 GHz and the data transfer rate is 54 Mbit/s . Figure 2.5 shows the WRT54GL specifications in a compact format. RJ-45 denotes the standardized physical network interface and is referred to registered jack 45. It is used to connect a device with a service provider i.e. the linksys WRT54GL is connected to the local internet provider via the RJ-45 network interface. This interface is also known as a twisted pair (CAT 5) cable. CAT 5 cables are used for structured cabling of computer networks in local area networks (LAN) i.e. for ethernet or fast ethernet data transmissions. The basic firmware of the linksys router does not support the Internet Protocol version 6 (IPv6). Therefore it is recommended to flash the initial firmware on the router and overwrite it with a firmware image as shown in Fig. 2.4 or OpenWRT.

Specifications

The specifications of the router is given in Fig. 2.5. The router is a wireless fidelity router with a Broadcomm 200 MHz central processing unit and a 16 MB Random Access Memory (RAM). The 4 MB Flash is implemented as an EEPROM (Electrically Erasable Programmable Read-Only Memory) and stores the firmware. The EEPROM is a non-volatile memory to store the data when the power supply is switched off. The antenna is implemented as a reverse polarity threaded Neill Concelman (RP-TNC) and the power supply is a 12 volts direct current (VDC).

2.4 Mathematical Principles

In classical information theory, the data transmitted can be interpreted as the content of a transmission between the source and the destination, where the content consists of a finite set of elements and this set of elements is a subset of a finite alphabet. As an alphabet, the classical bit values “0” and “1” can be taken. By contrast to classical information theory, it is necessary to distinguish the kind of the system, in which the transmission is done. By the use of a classical bit stream to transmit information between two communication parties, there exist various techniques to reconstruct the original message, if there are losses during the transmission. Quantum information theory focusses on the data and the interpretation (information), which is transmitted via particles (i.e. photons or light quanta). A communication system consists of a preparation device (the source) and a measuring device (the destination). That means from now on, it is necessary to focus on the length of the channel between the source and the destination. To specify the system, we want to distinguish the information from, we have to observe mathematical concepts. The essential information is given in this section, whereas the basics for it can be found in appendix A. The basic mathematic concepts are taken from [5, 14, 33].

Algebraic Structures

In abstract algebra, an algebraic structure is specified by a non-empty set of elements and a finitary operation on it. Finitary operations on a set of elements result in an element of the cartesian product of the set, mapped to the set itself. To define algebraic structures, we start with the definition of finitary operations.

Definition 2.1 (Finitary Operations) *A finitary, hence $n-$times operation φ in a set $\mathcal{A}$ is a function $\varphi : \mathcal{A}^n \to \mathcal{A}$, where every $n-$tuple of elements in $\mathcal{A}$ is assigned to an element from $\mathcal{A}$.*

We can specify this common definition for the case that $n = 2$ i.e. we define a binary operation for two elements a, b of the set $\mathcal{A}$ as follows.

Definition 2.2 (Binary Operations) *Finitary Operations with the case n=2 are special cases and will be indicated as binary operations e.g. addition or multiplication of numbers or matrices. Hence a binary operation is a function $* : \mathcal{A} \times \mathcal{A} \to \mathcal{A}$. For simplification, a binary operation is denoted by $a * b$ instead of $*(a, b)$. A binary operation $*$ in $\mathcal{A}$ is associative, if and only if*

$$(a * b) * c = a * (b * c)$$

and commutative, if and only if

$$a * b = b * a$$

$\forall a, b, c \in \mathcal{A}$, respectively.

An element $e \in \mathcal{A}$ is known as a neutral element respective to a binary relation $*$ in $\mathcal{A}$ if and only if

$$a * e = e * a = a \quad \forall a \in \mathcal{A}$$

holds.

An element $a^{-1} \in \mathcal{A}$ is an inverse element which satisfies

$$a * a^{-1} = a^{-1} * a = e \quad \forall a, a^{-1} \in \mathcal{A}$$

An n-fold sequential composition of the binary operation is defined as

$$a^n = \overbrace{a \circ a \circ \cdots \circ a}^{n-times}. \tag{2.1}$$

Thus, we can represent an n-fold repetition of an identical operation.

In some cases we want to investigate the elements $a, b \in \mathcal{B}$, where $\mathcal{B}$ denotes a subset of $\mathcal{A}$.

Definition 2.3 (Closure under Binary Operations) *Let $\mathcal{A}$ be a non-empty set of elements and let $a, b \in \mathcal{B}$ be arbitrary elements from the subset $\mathcal{B} \subseteq \mathcal{A}$. The set $\mathcal{B}$ has a closure under a binary operation if*

$$a * b \in \mathcal{B} \quad \forall a, b \in \mathcal{B}$$

holds.

That means that the set $\mathcal{B}$ is closed or has a closure under the binary operation, hence $a * b$ is in $\mathcal{B}$ if and only if the elements a, b are in $\mathcal{B}$. As an example, let $\mathcal{B} = \mathbb{N}_0$ and we interpret $I(*) = <$ as the less than operation. Hence the natural numbers are not closed under the less than operation, whereas the real numbers $\mathbb{R}$ are.

To achieve our goal, we need algebraic structures. An algebraic structure is an arbitrary set of elements with at least one finitary operation on it.

Definition 2.4 (Algebraic Structure) *Let $\mathcal{A}$ be a non-empty set of elements and let $\circ$ be a binary operation on $\mathcal{A}$ as a function $\mathcal{A} \times \mathcal{A} \to \mathcal{A}$ i.e. every pair of elements $a, b \in \mathcal{A}$ is mapped to an element from $a \circ b$. The pair $(\mathcal{A}, \circ)$ is called algebraic structure or a groupoid.*

That means the definition of an arbitrary algebraic structure prohibts the emptiness of $\mathcal{A}$. To define higher structures such as groups, we need algebraic structures.

Group Theory

After defining an algebraic structure, now we need knowledge about group theory and the common properties of groups, abelian groups and finite groups. Groups can be interpreted as algebraic structures with binary operations and common properties. This will be shown in this section.

Definition 2.5 (Semigroup) *Let $\mathcal{H}$ be a non-empty set with a binary operation $*$. Hence $\mathcal{H} = (\mathcal{H}, *)$ is called a semigroup if $*$ is associative. If a semigroup satisfies the commutative property, it is called an commutative semigroup.*

Classic examples for semigroups are $\mathbb{N}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ over the addition or the multiplication for instance. That means that a semigroup is an algebraic structure with the associative property holds. A monoid is an algebraic structure with an associative binary operation, hence it is a semigroup and the existence of an identity element. The result is the definition of a monoid as follows.

Definition 2.6 (Monoid) *Let $(\mathcal{M}, *)$ be an algebraic structure and let $\mathcal{M}$ be associative over $*$. $(\mathcal{M}, *)$ is called monoid if there exists a neutral element.*

The essential structure of this part is the group. It is specified as shown below.

Definition 2.7 (Group) *A set $\mathcal{G}$ with a binary operation $*$, denoted as $(\mathcal{G}, *)$ is a group, if $*$ is associative, closed under $\mathcal{G}$ and there exists a neutral element and for all elements $a \in \mathcal{G}$ exists an inverse element a^{-1}. Hence every element has exactly one inverse element in $\mathcal{G}$.*

That means that an algebraic structure is a group if it is associative, there exists a neutral element and for all elements exists an inverse element i.e. a group is a special semigroup. If a structure (semigroup, monoid, group) satisfies the commutativity property it is called commutative semigroup, monoid or group. Commutative groups are called abelian groups.

Definition 2.8 (Abelian Group) *Let $(\mathcal{G}, *)$ be a group. $\mathcal{G}$ is called abelian group, if $*$ is commutative.*

An abelian group is a special notation of a commutative group, in memory of the Norwegian mathematician Niels Henrik Abel (* 1802 – † 1829).

Definition 2.9 (Subgroup) *Let $(\mathcal{G}, *)$ be a group and $\mathcal{U} \subseteq \mathcal{G}$ be a non-empty subset of $\mathcal{G}$. If $\mathcal{U}$ is a group according to $*$, then $(\mathcal{U}, *)$ is a subgroup of $\mathcal{G}$. A non-empty set of of a group $(G, *)$ is a subroup of $\mathcal{G}$, if $a * b, a^{-1} \in \mathcal{U}$ holds $\forall a, b \in \mathcal{U}$. This can be denoted as $(\mathcal{U}, *) \leq (\mathcal{G}, *)$. If we set $\mathcal{U} = (\mathcal{U}, *)$ and $\mathcal{G} = (\mathcal{G}, *)$ it can be denoted as $\mathcal{U} \leq \mathcal{G}$.*

In other words, a subset $\mathcal{U}$ of a group $\mathcal{G}$ is a subgroup, if $\mathcal{U}$ forms a group under the binary operation $*$.

Definition 2.10 (Finite Group) *Let $(\mathcal{G}, *)$ be a group and $n \in \mathbb{N}_0$ be a non-negative integer. $\mathcal{G}$ is defined as a finite group iff $|\mathcal{G}| = n$, otherwise it is infinite.*

Rings and fields

In abstract algebra, the ring theory focusses on algebraic structures, which are called rings. Rings are algebraic structures where two binary operations, the addition and the multiplication are defined. We start with rings and fields to define the basis for a Hilbert space, which we need to understand the basic properties of quantum information. The definition of a Hilbert space is given in the next section.

In the set of integers ($\mathbb{Z}$), there exist at least two binary operations, "$+$" and "$\cdot$". We interpret $I\,(+)$ as the addition over the integers and $I\,(\,\cdot\,)$ as the multiplication over the integers. In ring theory, the basic concept is the ring as an algebraic structure with two binary operations. Its definition is given below.

Definition 2.11 (Ring) *Let $\mathcal{R}$ be a set with the binary operations "$+$" and "$\cdot$". An algebraic structure $(\mathcal{R}, +, \,\cdot\,)$ specifies a ring if the three following properties are satisfied*

$(\mathcal{R}, +)$ is a commutative group (with neutral element 0),
$(\mathcal{R}, \,\cdot\,)$ is a semigroup and
the distributive property holds as follows

$$\begin{aligned} a \cdot (b + c) &= a \cdot b + a \cdot c \\ (a + b) \cdot c &= a \cdot c + b \cdot c \end{aligned} \quad \forall a, b, c \in \mathbb{R}$$

That means the ring $(\mathcal{R}, +, \,\cdot\,)$ is an abelian group with an appended binary operation ("$\cdot$") that is associative and distributive over the abelian group operation.

Now we want to show the commutative property on a ring and specify a commutative ring as follows.

Definition 2.12 (Commutative Ring) *Let $(\mathcal{R}, +, \cdot)$ be a ring. If "$\cdot$" is commutative, then $(\mathcal{R}, +, \cdot)$ is a commutative ring.*

A commutative ring is defined similar to a commutative group as shown in the sections above.

The unity element or also called invertible element of a ring $\mathcal{R}$ is an arbitrary element u that has an inverse element on the multiplication of $\mathcal{R}$. $\mathcal{R}$ denotes a monoid as shown above. A formal definition of a unity and a neutral element is given in appendix A.

Definition 2.13 (Ring with Unity) *Let $(\mathcal{R}, +, \cdot)$ be a ring. If there exists a neutral element with respect to "$\cdot$", then $(\mathcal{R}, +, \cdot)$ is a ring with unity element.*

Definition 2.14 (Field) *Let $(\mathcal{F}, +, \cdot)$ be a commutative ring with unity element $1 \neq 0$ and every $a \in \mathcal{F}$ is a unit, hence an inverse element exists. Then $(\mathcal{F}, +, \cdot)$ is a field.*

Hence an algebraic structure $(\mathcal{F}, +, \cdot)$ is a field, if $(\mathcal{F}, +)$ and $(\mathcal{F} \setminus \{0\}, \cdot)$ are commutative groups and the distributive property holds. Hence the multiplication is associative, commutative, has a neutral element and for all elements in $\{\mathcal{F} \setminus \{0\}\}$ exists an inverse element. Now we have specified all the basic arithmetics, to construct the necessary Hilbert space for our further discussions.

Hilbert Space

The core point in this section is the definition of a Hilbert space. It was named after the German mathematician David Hilbert (* 1862 – † 1943). To define the Hilbert space, we need a vector space, the inner product and the complex and real numbers. To construct a vector space, we start with the definition of a vector as shown below. Further discussions on the inner product are given in chapter 4. The definition of the complex and the real numbers can be found in appendix A. The first geometric object we need is an euclidean vector, named after the Greek mathematician Euclid of Alexandria ($\approx$ 300 before christus). An euclidean vector has a magnitude and a direction. It is defined as follows.

Definition 2.15 (Euclidean Vector) *Let $\mathcal{F}$ be an arbitrary field. A column vector x in the $n-$dimensional space $\mathbb{R}^n$ is denoted by*

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

where the components are the coordinates of the vector $x_1, x_2, \ldots, x_n \in \mathbb{R}$.

A vector is an object that can be added to further vectors or it can be multiplied with a number. This number is called a scalar. A scalar is a mathematical quantity and is it is mapped to the real numbers in our case. An euclidean vector is an object in a vector space by definition. A vector space is specified by the definition as shown below.

Definition 2.16 (Vector Space) *Let $\mathcal{F}$ be a field and $(\mathcal{V}, +)$ be an abelian group. Hence the addition maps each pair (x, y) in $\mathcal{V} \times \mathcal{V}$ to $\mathcal{V}$. To all $\lambda \in \mathcal{F}$ and $x \in \mathcal{V}$ we assign the product $\lambda \cdot x \in \mathcal{V}$. This product is called the scalar product and maps each pair (λ, x) in $(\mathcal{F} \times \mathcal{V})$ or $(\mathcal{F} \times \mathcal{V})$ to $\mathcal{V}$. An algebraic structure $(\mathcal{V}, +, \mathcal{F})$ denotes a vector space or linear space over $\mathcal{F}$ if the following properties are satisfied for all $\lambda, \mu \in \mathcal{F}$ and $x, y \in \mathcal{V}$ and $\mathcal{F}$ is mapped to $\mathbb{R}$ or $\mathbb{C}$.*

$$\begin{aligned}
&x + y = y + x,\\
&x + (y + z) = (x + y) + z,\\
&\exists 0 \in \mathcal{V} \text{ s.t. } x + 0 = 0 + x = x,\\
&\exists\, (-x) \in \mathcal{V} \;\; \text{s.t.} \;\; x + (-x) = 0,\\
&\lambda \cdot \, (x + y) = \lambda \cdot x + \lambda \cdot y,\\
&(\lambda + \mu) \cdot x = \lambda \cdot x + \mu \cdot x,\\
&(\lambda\mu) \cdot x = \lambda \cdot \, (\mu \cdot x)\\
&1 \cdot x = x.
\end{aligned}$$

Definition 2.17 (A Norm on a Vector Space) *Let $\mathcal{V}$ be a vector space and let $\mathcal{F}$ be a field. A norm on a vector space is a function $\|.\|$: $\mathcal{V} \to \mathbb{R}$. The norm has to fulfill the following requirements for all $x, y \in \mathcal{V}$ and all scalars $\lambda \in \mathcal{F}$.*

$$\begin{aligned}
&\|x\| \geq 0,\\
&\|\lambda \cdot x\| = \lambda \cdot \|x\|,\\
&\|x + y\| \leq \|y + x\|,\\
&\|x\| = 0 \qquad\qquad \text{iff } x = 0.
\end{aligned}$$

The norm of a vector is commonly interpreted as the length of a vector. If a vector space has a norm, it is called a normed vector space or normed space.

Definition 2.18 (Unit Vector) *Let $x \in \mathcal{V}$ be a set of vectors with a norm. A vector x is called unit vector, if the vector length is 1, i.e. if $\|x| = 1$ holds. Let x be an arbitrary vector from $\mathcal{V}$ and let $\|x\|$ be its norm. Dividing the vector by its norm results in the unit vector as shown below.*

$$\left\| \frac{x}{\|x\|} \right\| = 1$$

Definition 2.19 (Metric) *Let $\mathcal{V}$ be a set and let d be a function as follows.*

$$d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$$

The function d is called a metric or distance function if d satisfies the following requirements for all $x, y, z \in \mathcal{V}$.

$$\begin{aligned}
&d(x,y) \geq 0, && \\
&d(x,y) = 0 && \text{iff } x = y, \\
&d(x,y) = d(y,x), && \\
&d(x,z) \leq d(x,y) + d(y,z). &&
\end{aligned}$$

Definition 2.20 (Metric Space) *A metric space is an ordered pair $(\mathcal{M}, d)$ where $\mathcal{M}$ is a set and d is a metric on $\mathcal{M}$.*

Now we want to specify a complete metric or Cauchy space. It is named by the French mathematician Augustin Louis Cauchy ($\star$ 1789 – † 1857).

Definition 2.21 (Complete Metric Space) *A complete metric is a metric $\mathcal{M}$ and every Cauchy sequence in it is convergent. Hence it has a limit that is also in $\mathcal{M}$.*

Definition 2.22 (Pre-Hilbert Space) *Let $\mathcal{V}$ be a vector space over the field $\mathcal{F}$. $\mathcal{V}$ specifies a pre-Hilbert space or a space with an inner product (scalarproduct) if for all pairs of elements $x, y \in \mathcal{V}$ are mapped to a number $(x, y) \in \mathcal{F}$. This number is called the scalar product of x and y. (x, y) denotes the inner product s.t. for arbitrary $x, y, z \in \mathcal{V}$ and an arbitrary $\lambda \in \mathcal{F}$ the following properties are satisfied. These properties are specified as the axioms of inner products.*

$$\begin{array}{ll}
H1 & (x,x) \geq 0 \\
H2 & (\lambda x, y) = \lambda (x,y) \\
H3 & (x+y, z) = (x,z) + (y,z) \\
H4 & (x,y) = \overline{(y,x)}
\end{array}$$

We can define a norm in a pre-Hilbert space by the scalar product as shown in the equation below.

$$\|x\| = \sqrt{(x,x)} \tag{2.2}$$

In the last step, we can define now a Hilbert space. A Hilbert space is specified by

Definition 2.23 (Hilbert Space) *Any pre-Hilbert space that is also complete is called Hilbert space.*

That means that a Hilbert space is a vector space with an inner product. It enables the measurement of the length and an angle of a vector. If the metric defined by the norm as shown above, is not complete, then $\mathcal{H}$ is called an inner product space.

2.5 Threats to Classical Cryptographical Methods

Quantum Computer

Classic computers operate on classical bit values, this is known as classical information. Also quantum computers operate on information but this information is different, it needs to be distinguished to the classical information. A quantum computer promises a speed up in the computation time over a classical computer. That means a quantum computer calculates a number of calculations simultaneously. It makes use of quantum mechanical properties to compute the result of a given problem faster than a classical computer. Because of the involvement of quantum mechanical properties it is not necessarily useful to use classical algorithms in a quantum computer alone. This introduces quantum algorithms which benefit from quantum mechanical phenomenons. Hence a quantum computer threatens to break classical cryptographic techniques because of its calculation power. Quantum algorithms are basic principles itself, because they are a reason why we need modern cryptographic techniques to protect information during the data transmission. The interested reader can find more detailed information on quantum computers in [12,35,37,38,49]. But we can already anticipate that a quantum algorithm can increase the speed of calculations in contrast to a classical algorithm.

2.6 Quantum Algorithms

A quantum algorithm takes n classical bits as input and creates a superposition of 2^n possible states. The quantum algorithm performs routines to obtain the result of the computation which can also be referred to as a quantum result. Finally a measurement is performed to obtain the n bits. The intermediate steps are combined into a single step. Quantum algorithms can be classified by their computation. Figure 2.6 shows an exemplary categorization of quantum

Computation	Algorithm		
	Deutsch Jozsa	Grover	Shor
Probabilistic		✓	✓
Deterministic	✓		
Finding Global Properties	✓		✓
Quantum Database Search		✓	✓
Brute Force		✓	✓

Figure 2.6: Categorization of Quantum Algorithms as Threats to Classical Computation

algorithms. Finding the global properties of a function by determining the function values, offers a decision whether a function is balanced or not. Hence a brute force method provides the desired result. Another type of algorithm solves the problem of searching in an unsorted database. Quantum database search is also known as the use of quantum algorithms. A brute force algorithm performs the systematically enumbering of all possible candidate solutions and checks whether a solution satisfies the problem. Furthermore, a combination of the previously mentioned methods as well provides a dedicated result.

Quantum computers profit from the benefits of quantum mechanical phenomena. By contrast to classical computation, phenomenons such as the quantum parallelism or the quantum interference, which does not exist in classical computational concepts, exist in quantum information theory.

Quantum parallelism is a phenomenon that can be described such as a quantum operation which is applied on a superposition of a set of input values, results in a superposition of a set of output values [4,38,49] i.e. a quantum computer is able to compute a finite number of operations simultaneously. The quantum parallelism was first described by David Deutsch in [11]. It was specified to distinguish the parallelism in quantum computations to the parallelism in classical computations. Suppose a classic computer with a multicore processor i.e. a processor with n independent central processing units (CPUs) to read and write program executions. These CPUs are linked together to execute different instructions simultaneously. That means if we need an exponential grow up in classic computation power, we need an exponential grow up in CPUs. By contrast, suppose a single quantum central processing unit, which is able to perform the computation of n instructions simultaneously. That means that an exponential grow up in the computation power of a quantum computer does not need necessarily an exponential grow up in quantum central processing units i.e. the computation power increases with the number of used quantum bits. This phenomenon is possible because a quantum bit or qubit exists in several states simultaneously and this is called a superposition. A superposition is an overlay of the same physical quantities in a system. It will be specified more in detail in chapter 4 at the beginning, because it is not in the scope of the concepts of quantum algorithms.

Quantum interference is a phenomenon, which is necessary for quantum parallelism. It is the interference of quantum mechanical objects i.e. the wave propagation of light quantas. Quantum objects behave as waves and particles which is formally known as the wave-particle duaslism. That means a quantum object propagates as a wave on the channel but it can only be measured as a particle at a specific time. More information on quantum interference is given in [27,35,37,38,49].

CHAPTER 3

Internet Protocol

The internet protocol (IP) is an open systems interconnection (OSI) network layer (layer-3) protocol suite. IP is used to solve the problems of packet forwarding and routing across computer networks from one endpoint to another one. It is specified in request for comments (RFC) 760 (1980) and RFC 791 (1981) by the Internet Engineering Task Force (IETF). The IP version 4 (v4) offers a 32 bit address space. Its address exhaustion is evolving due to the new growing markets in the BRICS (Brazil, Russia, India, China, South Africa) countries, an increasing number of users of mobile technologies or the distribution of resources over wide areas. The growth of the internet in recent years has led to an evident lack of unallocated addresses. IPv6 increases the number of bits for the source and destination address up to 128. Hence it can solve the evident IPv4 address problem and replace the IPv4 internet with at most 2^{128} unique addresses.

3.1 IPv4 Specification Overview

The OSI Internet Protocol Version 4 (IPv4) is the first globally implemented and commonly used protocol to standardize the communication on the internet. Fig. 3.1 shows the models of the department of defense (DOD) and the open systems interconnection model (OSI). IPv4 is used for the transmission of data packets between communication parties over a connectionless, packet oriented network. The main idea was that every communication endpoint is identified by a unique address in the network and specifies as an end-to-end connectivity. Due to the lack of addresses, various techniques evolved to increase the number of advertized hosts in a network, such as the network address translation (NAT). NAT is a solution to delay the deficiency because of its 16-bit port number field design, with at most 2^{16} simultanous connections from one public IP address to internal nodes. Particularly, NAT contradicts the end-to-end pattern but is not in the scope of this work. It causes delays and does not support all applications, such as asynchronous full transfer zone (AXFR).

OSI	DoD
Application Presentation Session	Process/Application
Transport	Host to Host
Network	Internet
Data Link Physical	Network Access

Figure 3.1: Open Systems Interconnection Model (OSI) and Department of Defense layer Model (DOD)

Host Numbers - Domain Name System

Figure 3.2 gives an overview of the numbers of hosts, advertised in the DNS between 1981 and 2012. It shows that the number of hosts steadily grows and that the free address space decreases. The IPv4 design is based on recommendations and a design process from the early 1980s. In 1981 the number of advertised hosts to the domain name system (DNS) was 213. Hence a 32 bit (8-nibble) address space of at most $2^{32} = 4\,294\,967\,296 \approx 4,29 \cdot 10^9$ unique addresses was deemed to be sufficient for an adequate time. In July 2012 the number of advertised hosts to the DNS was 908 585 739 and the total number of theoretically available, unique addresses can no longer meet these demands. More information on IPv4 and the DNS can be found in [1, 18, 47].

Date		Survey Host Count
2012	Jul	908 585 739
	Jan	888 239 420
2011	Jul	849 869 781
	Jan	818 374 269
2010	Jul	768 913 036
	Jan	732 740 444
1993	Jul	1 776 000
	Jan	1 313 000
1981	Aug	213

Figure 3.2: Number of Hosts advertised in the Domain Name System (DNS) - [6, 7]

IPv4 Header

An IPv4 packet has at least 13 and at most 14 fields of information. A simplified representation is given in Fig. 3.3. The field of IP options is not necessary. However, it can be necessary to

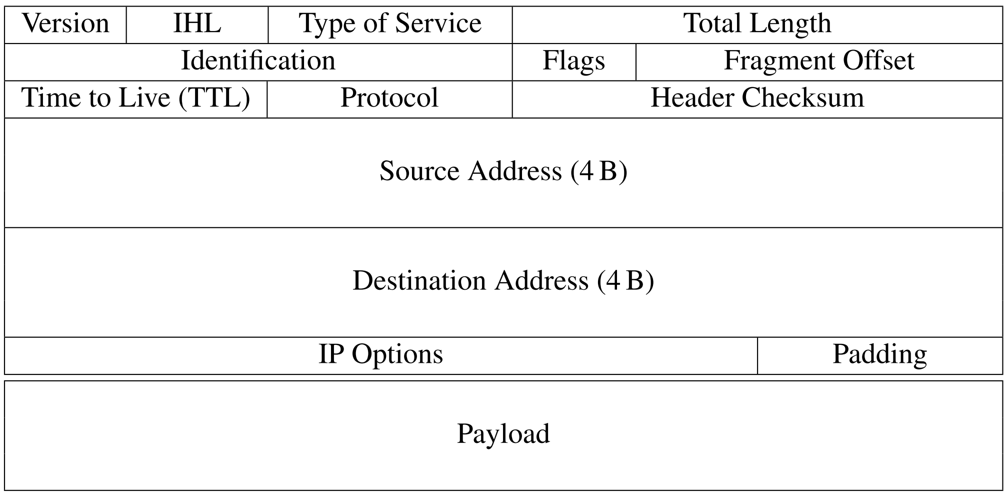


Figure 3.3: Basic Header + Payload of IPv4

add further information to the IP header. Two main applications for additional information in the options field are pre-defined routes or statistical analysis. The options field is appended after the required header fields but it preceedes the payload (data to be transmitted). An IPv4 packet starts with a 4 bit Version field, where `0100` denotes the version 4 pattern. The 4 bit internet header length (IHL) field specifies the length of the IP packet header. Including the options field and padding. The next field is the 8 bit type of service (ToS) field. It provides quality of service (QoS) features such as delay, precedence (prioritized delivery), reliability and throughput. The total length (TL) of an IP packet is specified by the so called 16 bit field in the header. The TL field consists of the IP header information and the payload. The lower bound for an IPv4 packet is represented by an IP header without payload (length of payload is 0), hence the minimum length of an IPv4 packet is the minimum length of the IP header (20 B). The upper bound is $2^{16} = 65\,536 = 64$ kB, based on its 16 bit length. An IPv4 packet consists of an identification field. This field is used to number consecutive parts of data in case it has to be split between several packets. Hence every multiple packet is assigned to the same identification value and there is no risk of mixing the fragments, received by another fragmented flow (segmentation and reassembly). If a flow needs to be fragmented, the flag field indicates the set of information (reserved, DF, MF). The first control flag (reserved) is a bit which is assigned to “0” (invariant) and is not used. The second control bit (DF, do not fragment) is “0” if the flow is fragmented, or “1” if it is not fragmented. The third flag is “0” if no more fragments need to be sent (or the original flow never was fragmented) or “1” else. The offset, where the fragmented data in the packet is located, is specified by an 8 B fragment offset field. The first fragment has offset 0. The TTL (time to live) specifies the maximum time (or number of router hops), how long an IPv4 packet can survive on a network (router hops) before it is discarded. All involved routers decreese this number by 1 or the number of seconds, delayed by the calculations in the router. If the number in this field is 0, the packet is discarded. The next field is the 8 bit protocol

Class	Leading Bits	Number of Networks	Number of Hosts
A	0	2^7	2^{24}
B	10	2^{14}	2^{16}
C	110	2^{21}	2^8
D	1110	28 bit Multicast groupID	
E	11110	28 bit reserved for experimental tasks	

Figure 3.4: Network Classes, Numbers of Networks and Hosts of IPv4

field. It specifies the protocol that receives the dedicated payload. The 16 bit header checksum field is computed over the entire header to provide protection against corruption and to verify the validity of the IPv4 header by a 16 bit checksum. Every intermediate router calculates the checksum, verifies its validity and if it is corrupted, the IPv4 packet is discarded. The source address is denoted as a 32 bit, invariant IP address of the sender of the original flow. Intermediate devices do not manipulate this address. The second invariant 32 bit address is the destination address. It is invariant for intermediate devices and specifies the intended recipient of the original packet. Particularily, the IP options field denotes settings for debugging, security or testing. The header is padded with 0's to make the header length a constant length of 32 bit.

Addressing

The length of an arbitrary IPv4 address is 32 bit. Its structure is specified by binary streams `b3.b2.b1.b0` where $b_i \in [0000\,0000, \ldots, 1111\,1111]$ denotes the i^{th}-byte. This corresponds to the decimal digits `d3.d2.d1.d0` where $d_i \in \{0, \ldots, 255\}$. The address is represented by 4 decimal digits each mapped to an 8 bit octet, respectively. The address 127.0.0.1 denotes the loopback address. It allows the local machine to refer to itself. Nevertheless the number of IPv4 addresses are not sufficient for future demands. Hence we need a scalable solution to solve the imminent problem of the lack of IPv4 addresses. Another problem is the dissipation of network fragments by predefined, rigid network classes for unicast addresses (A, B, C) and multicast addresses (D) with fixed size of hosts. Class E networks are reserved for experimental tasks. An overview is given in Fig. 3.4. A newer version of IP is necessary to provide a bigger address space.

3.2 IPv6 Specification Overview

The requirements concerning the increased address space, an adequate modularity and an integration of security features resulted in a simplified header format. The development of IPv5 was aborted. The development of IPv6 was started in the 1990's by the Internet Engineering Task Force (IETF) [26]. IPv6 permits to address $2^{128} \approx 3.4 \cdot 10^{38}$ devices which exceeds the IPv4 address space by 29 orders of magnitude to avoid the necessity of NAT which violates the point to point pattern.

3.3 IPv6 Header

The IPv6 header consists of a new header format with reduced complexity, an efficient method of hierarchical addressing and routing, native integrated security mechanisms and extensibility by extenion headers. The reduced complexity in the header format results in a minimalization of the data to be processed by routers. To provide sufficient scalability in an IPv6 packet, the header was constructed without containing any optional elements. Hence, this fixed header format provides more efficiency than in IPv4 which results in an increased performance. Instead of optional elements in the header, extension headers are introduced by IPv6. Furthermore, the header checksum has been abandoned which results in an increased performance, too.

Basic Header

The basic IPv6 header without extensions is shown in Fig.3.5. The version field in the IPv6 header contains `0110` signifying version 6. Hence this unique identification allows us to mix IPv4 and IPv6 traffic in the same network. The 8 bit traffic class field (packet priority) is similar to the type of service field from IPv4 and allows us to distinguish different levels of priority of IPv6 packets. It does this by QoS information [34]. Flow label (QoS management) denotes a 20 bit field to distinguish 2^{20} different traffic flows. It identifies a set of packets belonging to a specific flow. A flow is a sequence of packets sent from a source to a destination. Hence an intermediate router can identify the packets with equal traffic flow values and handle them identically. Another important field is the payload length. It has 16 bit and denotes the packet length of the payload. In contrast to IPv4, the length of the basic header is not included. However, the length of any extension header is included. To prevent an arbitrary packet indefinitely running through a network, the hop limit field (8 bit) is part of the header. Every intermediate device i.e. router decrements the hop count by 1. If the hop count limit reaches zero, the packet is discarded. The hop-by-hop limit is similar to the TTL from IPv4. The next header value is UL if a packet is sent without extensions and specifies the upper layer protocol i.e. TCP, UDP or ICMPv6 and the payload is appended without a subsequent extension header. The 128 bit source address field contains the IPv6 address of the originator whereas the destination field (also 128 bit) contains the IPv6 address of the receiver. Both addresses are invariant for intermediate nodes during the transmission on the network. If no extension header is specified, the payload is appended after the destination address.

Extension Header

One of the weaknesses in IPv4 is the lack of scalability and the security mechanisms. By contrast, the implementation of security mechanisms in IPv6 was part of the initial design process. The structure of an extension header is shown in Fig. 3.6. It shows its length, the next header value, the options and the padding to fill missing 0s. A simple representation of the basic IPv6 header packet and appended extension headers plus payload is given in Fig.3.7. To specify the next header, there exists an 8 bit field in IPv6. It specifies the immediate succeeding header to protect against corruption. After the basic (main) header is completed and the specification of all extension headers is done, the payload is appended.

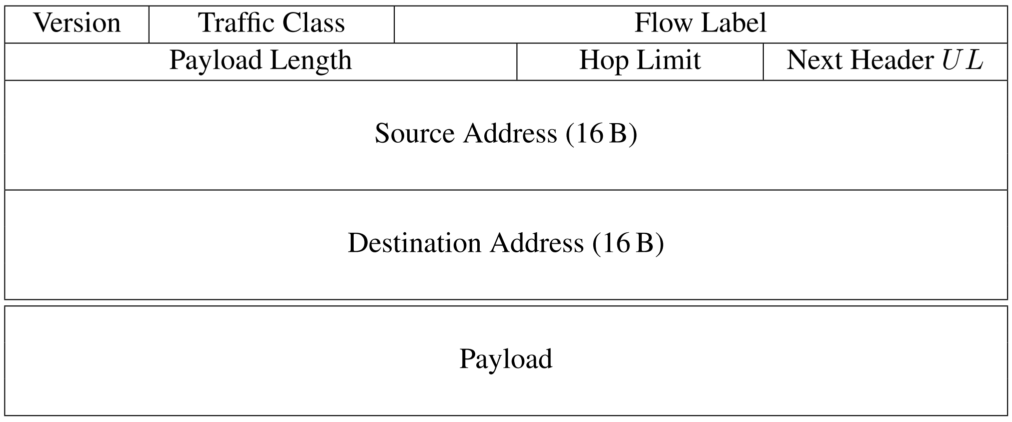


Figure 3.5: Basic Header + Payload of IPv6

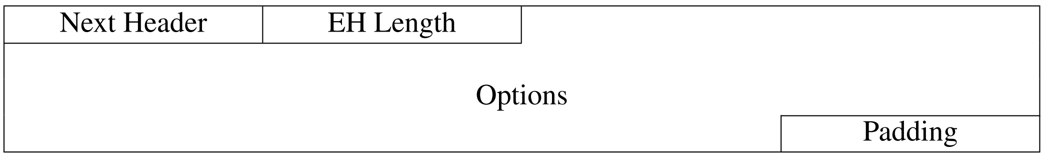


Figure 3.6: Basic Extension Header of IPv6

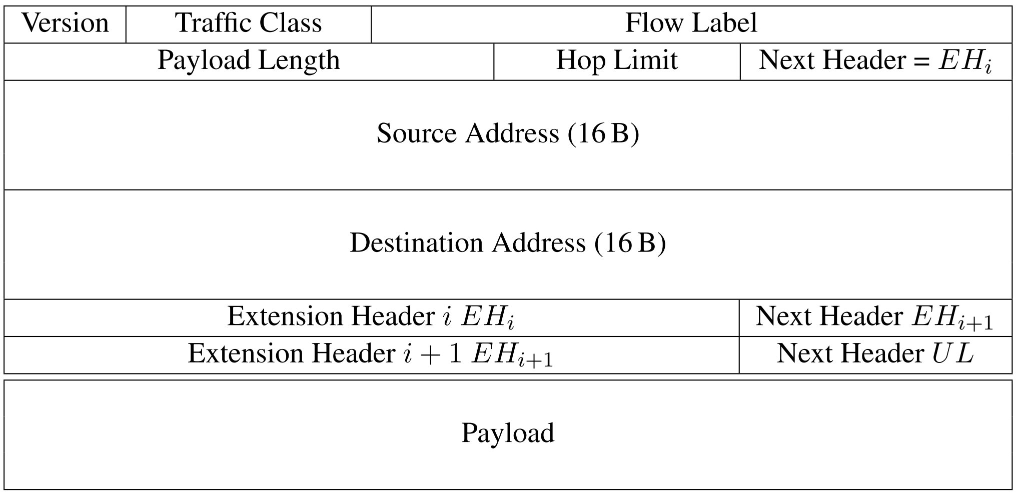


Figure 3.7: Basic Header + Extension Headers + Payload of IPv6

Order	Header Type	NH ID (DEC)	NH ID (HEX)	size
1	Basic IPv6 Header	-	-	8 octets
2	Hop-by-Hop EH	0	00	var.
3	Destination Options	60	3C	var.
4	Routing EH	43	2B	var.
5	Fragment EH	44	2C	8 octets
6	Authentication EH	51	33	var.
7	Encapsulated Security Payload EH	50	32	var.
8	Destination Options EH	60	3C	var.
9	Mobility EH	135	87	var.
	No next Header exists	59	3B	empty
UL	TCP	6	06	
UL	UDP	17	11	
UL	ICMPv6	58	3A	

Figure 3.8: Default IPv6 Headers

Fig. 3.8 shows the default extension headers (next header NH) and the corresponding decimal, hexadecimal and order value. The architecture of an optional, default extension header is shown in Fig. 3.6. Every extension header from Fig. 3.8 has the same architecture. The order of extension headers from Fig. 3.8 is recommended to use, but is not mandatory. The use of the ESP header without authentication is possible, but not necessarily useful. Instances of Fig. 3.7 are shown in Fig. 3.9. It lists examples of extension headers and the corresponding extension header concatenation using the recommended order in Fig. 3.8. The basis concatenation for a valid TCP transmission is the IPv6 basic header and the TCP header (06_{16}). After that, the payload is appended. An extension with the routing header $(2B_{16})$ enables a connection between two communication parties over a network. A routing header extension must be the same in all packet flows. A flow is a sequence of packets. If a router receives a packet of a new flow the router calculates the result of the carried information, caches it and re-uses it for every incoming packet with the same flow. This method speeds up the calculation for all packets in the same flow. Flow fragmentation by routers is not implemented in IPv6. If there exists a link with a maximum transmission unit (MTU) than the original size of the IPv6 packet on a route between two communication parties, the flow needs to be fragmented, anyway. In IPv6, the original flow is split into fragments by the sender. The sender transmits an arbitrary set of packets with an arbitrary length. It checks, if the packets are received correctly. Oversized packets are returned as undeliverable and the router replies with a statement that fragmentation is required. The fragmentation header $(2C_{16})$ specifies the parameters by the sender. Without a routing header extension, the fragmentation header extension is not necessary. To transmit a large payload >64 kB, the hop-by-hop extension header $(3C_{16})$ specifies the corresponding parameters. That means if a router has a MTU e.g. 128 kB and the initial flow was e.g. 256 kB we need the hop-by-hop and fragmentation extension headers, respectively. If the hop-by-hop extension header is part of an IPv6 packet, it must be the first one, appended after the basic header. The header

IPv6 Basic Header Next Header = 06	TCP Header + Payload		
IPv6 Basic Header Next Header = 2B	Routing Header Next Header = 06	TCP Header + Payload	
IPv6 Basic Header Next Header = 2B	Routing Header Next Header = 2C	Fragment Header Next Header = 06	TCP Header + Payload
IPv6 Basic Header Next Header = 00	Hop-by-hop Next Header = 3C	Destination Options Next Header = 06	...

Figure 3.9: Header concatenation and the IPv6 Modularity

IPv6 Basic Header	plain(data)
Extension Header	
ESP Header	
enc(payload)	encrypted(data)
auth(data)	

Figure 3.10: Structure of the Security Header of IPv6

modularity from IPv6 based on the ESP is shown in Fig. 3.10. The interface between encrypted data and plain data is the ESP header. That means that the IPv6 basic header, the extension header values and the esp header are transmitted as a plaintext, respectively. The payload and the authentication values in the packet are transmitted as a ciphertext. More information on the extension headers in IPv6 can be found in [9, 34, 47].

3.4 IPv6 Adressing

The address space increased from 32 bit in IPv4 to 128 bit by IPv6. Extending the notation used for IPv4 addresses to IPv6 results in strings of the form

Address Type	IPv6 Address Instance
Expanded:	`2001:0db8:85a3:08d3:1319:8a2e:0370:7347`
Eliminating leading 0s:	`2001:db8:85a3:8d3:1319:8a2e:370:7347`
Expanded:	`0000:0000:0000:0000:0000:0000:0000:0000`
Eliminating leading 0s:	`0:0:0:0:0:0:0:0`
Eliminating a sequence of 0-blocks:	`::`
Expanded:	`0000:0000:0000:0000:AC43:9876:1234:5555`
Eliminating leading 0s:	`0:0:0:0:ac43:9876:1234:5555`
Eliminating a sequence of 0-blocks:	`::ac43:9876:1234:5555`
Expanded:	`3228:00A3:0000:0000:0221:01FF:CE01:7842`
Eliminating leading 0s:	`3228:A3:0:0:221:1FF:CE01:7842`
Eliminating the first sequence of 0s:	`3228:A3::0:221:1FF:CE01:7842`
Eliminating concatenated blocks of 0s:	`3228:A3::221:1FF:CE01:7842`
Expanded:	`4ac0:0000:0000:0007:0000:0000:0000:000c`
Eliminating leading 0s:	`4AC0:0:0:7:0:0:0:C`
Eliminating the first sequence of 0s:	`4AC0::7:0:0:0:C`
Eliminating the second sequence of 0s:	`4AC0:0:0:7::C`

Figure 3.11: Exemplary Overview of IPv6 Addresses

`d15.d14.d13.d12.d11.d10.d9.d8.d7.d6.d5.d4.d3.d2.d1.d0` where the d_i are numbers in the range between 0 and 255. To improve readability, IPv6 addresses are written in hex notation, with colons offer every four digits, respectively. Leading zeros in these four digit blocks can be omitted.

If a sequence of blocks is zero, it can be shortened to two colons (::). To avoid ambiguities, only one such sequence may be abbreviated this way. Figure 3.11 demonstrates the possible simplifications for IPv6 addresses. More detailed information on IPv6 and the address formats can be found in [1, 9, 10]. IPv6 distinguishes 3 different modes, unicast, multicast and anycast. The types of addressing are explained in the following sections.

Unicast

An unicast addressing method is specified by a unique identifier for a single interface. A bijective function between two communication parties maps one interface endpoint in the network uniquely to another. Hence a packet is delivered to the interface, identified by the unicast address. Figure 3.12 shows the bijective function between the sender $\mathcal{H}_0$ and the receiver $\mathcal{H}_1$. The IPv6 addressing architecture by the IETF [10] specifies two principal allocated types of unicast addresses (Globally Unique Addresses, Unique Local Addresses) [8]. A globally unique address identifier is a unique 128 bit address. It is separated into a public and a location specific part. The public section transmits the routing information and the location specific section transmits the structure of the dedicated subnet. The unique local address denotes a private section as a part of a private location subnet i.e. a private network inside an organization or as a part of a site. More detailed information on unicast addresses can be found in [1, 23, 24]

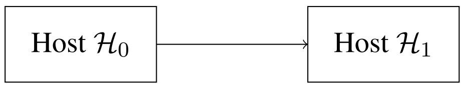


Figure 3.12: Unicast Addressing of IPv6

Multicast

Multicast addressing is an IP packet delivering method to transmit one packet to multiple nodes in an unicast group as shown in Fig. 3.13. Only one transmission is required. This IPv6 technique ensures that a packet, sent to a multicast address, is delivered to all connected interfaces in a multicast group. More information about multicast addressing can be found in [1]

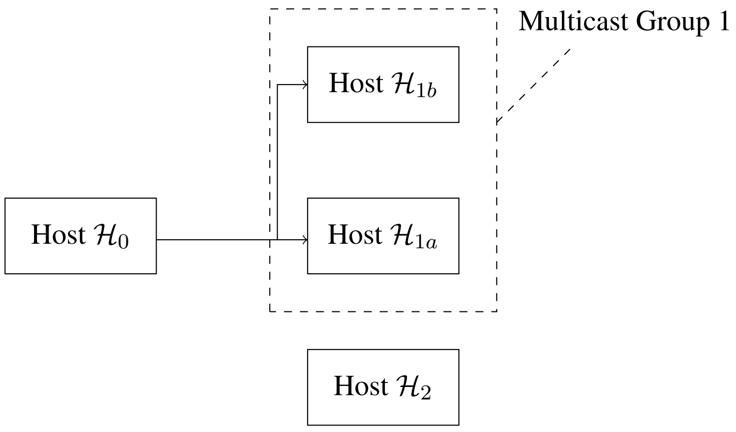


Figure 3.13: Multicast Addressing of IPv6

Anycast

Anycast addresses are similar to unicast addresses except that one address instance is mapped to a group of interfaces (hosts) consisting of at least one element (Fig. 3.14). In contrast to a multicast or unicast address, an IP packet is sent to an arbitrary interface identified by the anycast address. It will be delivered to an available or an arbitrary interface, chosen by a router, routing protocol or load balancer. Choosing the nearest interface according to some ranking metrics enables failover. By setting an anycast flag, multiple unicast address error messages will be prevented. More information about anycast addresses can be found in [1, 19]

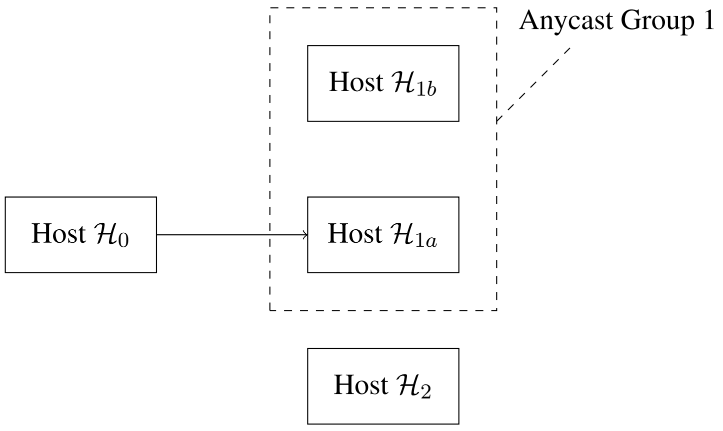


Figure 3.14: Anycast Addressing of IPv6

3.5 Securing IPv6 by IPsec

The internet protocol security (IPSec) is a set of protocols and algorithms to implement security features in IPv6. The main specification is given in [31] and was developed by the IETF. According to the required security level, IPSec allows two communication parties to select the security algorithms, the protocols and the authentication method. IPSec provides two primary security concepts, the authentication header (AH) and the encapsulating security payload (ESP).

Authentication Header

The specification of the authentication header (AH) is given in [30]. The authentication header is appended after the IP header and protects against resend attacks and insertion of manipulated packets. This is accomplished by ensuring data integrity and authenticity by cryptographic hash functions. Data integrity means that the data remains unchanged during transmission. Authenticity means that the sender can be identified beyond doubt. However, the payload is transmitted as a plaintext and can be read by everyone. For this reason, the authentication header is rarely used.

Encapsulating Security Payload

The encapsulating security payload (ESP) [29] provides methods to implement the key security concepts authenticity, confidentiality and integrity. The main difference between ESP and AH is that the payload is transmitted encrypted on the public channel. Moreover, ESP provides encryption-only and authentication-only modes with the first one being not particularly useful. The first implemented security algorithm is the `ESP-NULL`, which does nothing and implements the authentication and integrity without confidentiality. Another implemented security mechanism is the Diffie Hellman Key exchange technique. The other implemented security mechanisms are not in the focus of this work. The interested reader can find more information about the symmetric encryption algorithm DES (Data Encryption Standard), 3DES (Triple

DES), AES (Advanced Encryption Standard), or the symmetric block ciphers Blowfish, CAST-128, IDEA (International Data Encryption Algorithm), RC5 (Rivest Cipher 5) and the hash function RIPEMD-160 (RACE Integrity Primitives Evaluation Message Digest) in [1,42,46]

Internet Key Exchange

The internet key exchange (IKE) [20,28] is a management and key generation protocol in IPSec that sets up security associations (SA). A security association is a collection of parameters for establishing a secure communication between two communication parties. IKE is based on the Diffie Hellman key exchange to establish a shared secret session key i.e. a cryptographic key only used for one session between a source (sender) and a destination (receiver). The Diffie Hellman Key exchange (or key agreement protocol) was first proposed in 1976 by Whitfield Diffie and Martin E. Hellman in [13]. It uses an arbitrary unsecure channel for communication, like a classic internet connection without security mechanisms. Its security depends on the difficulty to compute discrete logarithms of large numbers. Suppose that two distant communication parties want to share a secret key K and Eve listens to the unsecure channel to gain information. The goal of the Diffie Hellman key agreement protocol is to transmit information over the unsecure channel that the key K is not available to Eve i.e. to generate a random shared secret between two remote communication parties, which are connected via an unsecure channel e.g. the internet. They can use the shared secret to exchange confidential data over the unsecure channel. The key can be used as a session key or it can be reused or it can be used only once. The Diffie Hellman key exchange provides only the key. The data exchange between the communication parties allows to draw conclusions only with a great time and effort to an eavesdropper. To explain the Diffie Hellman key exchange we specify the order of an integer.

Definition 3.1 (Order) *Let $n \in \mathbb{N}$ and $x \in \mathbb{Z}$ such that*

$$a^x \equiv 1 \bmod n$$

holds. The order of a modulo n is the smallest positive integer x and it is denoted by $ord_m(a)$.

The Euler's totient function provides that such an integer exists. It is named by the Swiss mathematician Leonhard Euler (1707 – †1783) and it is given below.

Definition 3.2 (Euler's totient function) *Let $n > 1$ be a positive integer. The Euler's totient function counts the number of positive integers less than or equal to n that are coprime. It is denoted by $\varphi(n)$.*

That means it counts the number of integers k in the range $1 \leq k \leq n$ for which $gcd(n, k) = 1$ holds. Euler's totient function has two primary properties. The first is that for an arbitrary prime number p, $\varphi(p) = \varphi(p-1)$ holds since all numbers which are smaller than p are coprime to p. The latter property is that for two coprime numbers m, n where $gcd(m, n) = 1$ holds it is multiplicative i.e. $\varphi(m, n) = \varphi(m) \cdot (\varphi)$. We substitute $[x \mid t]$ and $[t \mid \varphi(n)]$. That means

$$a^x \equiv a^{\varphi(n)} \equiv 1 \bmod n \tag{3.1}$$

holds.

Definition 3.3 (Primitive Root) *The integer a is called a primitive root if*

$$\varphi(n) = ord_n(a)$$

holds.

Let p, q be two numbers s.t. p is a prime number and q is a primitive root and p, q are exchanged over the unsecure channel. The secret key K_S is specified by the logarithm of the private key K_P to the basis $q \bmod p$ for Alice and Bob, respectively.

$$\begin{aligned} K_{S,A} &= \log_q(K_{P,A}) \bmod p \quad \text{s.t. } 1 \leq q \leq p-1 \\ K_{S,B} &= \log_q(K_{P,B}) \bmod p \quad \text{s.t. } 1 \leq q \leq p-1 \end{aligned} \tag{3.2}$$

To construct an instance for the public key $K_{P,A}$ Alice choose a random secret key $K_{S,A}$ and Bob choose $K_{S,B}$ to construct its public key $K_{P,B}$ as shown below. Alice and Bob choose a large random number to construct an instance of the secret key, respectively.

$$\begin{aligned} K_{P,A} &= q^{K_{S,A}} \bmod p \\ K_{P,B} &= q^{K_{S,B}} \bmod p \end{aligned} \tag{3.3}$$

The communication partners A and B exchange the computed values $K_{P,A}, K_{P,B}$ as a plaintext on the public channel, respectively. Alice computes the key K_{AB} and Bob computes the key K_{BA}.

$$\begin{aligned} K_{A,B} &= (K_{P,B})^{K_{S,A}} \bmod p \\ K_{B,A} &= (K_{P,A})^{K_{S,B}} \bmod p \end{aligned} \tag{3.4}$$

The keys $K_{A,B}$ and $K_{B,A}$ from Eq. (3.4) are equal because of the rules for exponentiation and associativity, the equation

$$\left(q^{K_{S,A}}\right)^{K_{S,B}} = \left(q^{K_{S,B}}\right)^{K_{S,A}} = q^{\left(K_{S,A} \cdot K_{S,B}\right)} \tag{3.5}$$

holds. This key can be used to establish a secure connection between two communication parties i.e. $K = K_{A,B} = K_{B,A}$ is the shared secret key. The computation steps from the Diffie Hellman key agreement protocol can be summarized to secret and public parameter calculations and a public parameter exchange between A and B. The protocol rounds are shown in Fig. 3.15.

Public Parameter Calculation	
p, q	s.t. p is a prime number and q is a primitive root of p
Secret Parameter Calculation	
A	B
choose $K_{S,A}$	choose $K_{S,B}$
$K_{P,A} = q^{K_{S,A}} \bmod p$	$K_{P,B} = q^{K_{S,B}} \bmod p$
Public Parameter Exchange	
A $\longrightarrow K_{P,A} \longrightarrow$ B	
A $\longleftarrow K_{P,B} \longleftarrow$ B	
Secret Parameter Calculation	
A	B
$K_{A,B} = (K_{P,B})^{K_{S,A}} \bmod p$	$K_{B,A} = (K_{P,A})^{K_{S,B}} \bmod p$
$K_{A,B} = (K_{P,B})^{K_{S,A}} \bmod p = K = (K_{P,A})^{K_{S,B}} \bmod p = K_{B,A}$	

Figure 3.15: Protocol Rounds of the Diffie Hellman Key Agreement Protocol

To show an example, we interpret the values as follows.

$$p = 17,\ q = 12,\ K_{S,A} = 18,\ K_{S,B} = 5.$$

Referencing to Eq. (3.3), A and B choose $p = 17$ as a prime number and $q = 12$ as a primitive root from 17 (one can choose alternatively the primitive roots $3, 5, 6, 7, 10, 11, 14$). The values 18 and 5 are kept secret, respectively.

$$\begin{aligned}
K_{P,A} &= 12^{18} \bmod 17\\
&= 26\,623\,333\,280\,885\,243\,904 \bmod 17\\
&= 8 \bmod 17\\
&= 8\\
K_{P,B} &= 12^{5} \bmod 17\\
&= 248\,832 \bmod 17\\
&= 3 \bmod 17\\
&= 3
\end{aligned}$$

Alice and Bob, now exchange the computed values 8 and 3, respectively and compute the final key in the next step.

$$\begin{aligned} K_{A,B} &= 3^{18} \bmod 17 \\ &= 387\,420\,489 \bmod 17 \\ &= 9 \bmod 17 \\ &= 9 \\ K_{B,A} &= 8^{5} \bmod 17 \\ &= 32\,768 \bmod 17 \\ &= 9 \bmod 17 \\ &= 9 \end{aligned}$$

As a result, Alice and Bob get the key $K_{A,B} = 9$ to initiate their secret communication i.e. the shared secret key in our calculation is 9. From Eq. (3.5) it follows that $K = K_{A,B} = K_{B,A}$ holds in the computation. An overview of secret and public parameters in the Diffie Hellman key agreement protocol is given in Fig. 3.16. A public parameter is known by everyone and a secret parameter is assumed to be confidential. The computations shown above are known as the

Parameter	Secret	Public
p		✓
q		✓
$K_{S,A}$	✓	
$K_{S,B}$	✓	
$K_{P,A}$		✓
$K_{P,B}$		✓
$K_{A,B}$	✓	
$K_{B,A}$	✓	
K	✓	

Figure 3.16: Secret and Public Parameter Overview Diffie Hellman Key Agreement Protocol

Diffie Hellman problem (DHP) or the Discrete Logarithm Problem (DLP). The basic arithmetics of groups can be found in appendix A. Its definition is given below.

Definition 3.4 (Discrete Logarithm Problem) *Let $(G, *)$ be an abelian group where $*$ denotes the group operation. We define the Discrete Logarithm Problem (DLP) as to determine an integer x for arbitrary given elements $g, h \in G$ s.t. x satisfies the following condition.*

$$\overbrace{g * g * \cdots * g}^{x \text{ times}} = h$$

Its difficulty depends on the specifications of the group G. The smallest value of x is called the discrete logarithm of h with basis g. The discrete exponentiation is easy to compute, whereas the discrete logarithm is harder to compute than a classic logarithm function.

The DHP is a problem that most cryptographic systems are based on mathematical functions that are easy to compute but computationally hard to reverse. It is defined as follows.

Definition 3.5 (Diffie Hellman Problem) *Let p be a prime number and g be an arbitrary integer. The Diffie Hellman Problem (DHP) is the mathematical problem of computing the value $q^{xy} \bmod p$ from the values $q^x \bmod p$ and $q^y \bmod p$ and p.*

Logarithms in real numbers are computational easy because they are continous and monotonically increasing functions. A discrete logarithm does not have the mentioned properties. That means that the DHP is at most as hard as the DLP; if one can solve the DLP, the DHP can be solved. More information on computational complexity can be found in [13, 41].

Security Problems in the Face of Quantum Computing

A cryptographic system is a protocol suite of security features to implement an encryption and decryption system in the field of data processing. Cryptographic systems are used to secure data transmission between distant nodes and establish a secure communication to protect against eavesdroppers. It is defined as a secure cryptographic system if it is impossible (note the difference between impossible and hard to compute) to encrypt the ciphertext and obtain the plaintext without knowledge on the key. The core security components of IPv6, such as the Diffie Hellman key exchange method, are based on unproven complexity assumptions. That means, if we have infinite time and infinite computing power, cryptographic systems based on these unproven computational hardness assumptions may be broken. Hence, the question is, how long will classic cryptography be secure? The increasing number of internet users in recent years and the data processing in various industrial sectors and private communications results in the demand to secure communication over the internet. The main potential threats are increasing computing power, the enhancement on mathematics and quantum computers. Quantum computers are potential threats to asymmetric, public key cryptography. They make use of quantum algorithms such as the Deutsch Jozsa algorithm to find global properties of functions and quantum data base search algorithms based on brute force attacks, such as the Grover and Shor algorithm as shown in Fig. 2.6.

Quantum key distribution (QKD) is a promising method for securing communication for future demands in information technology and is often referred to as quantum cryptography (QC). QKD is a method to generate a shared random string between two communication parties which can be used as a secret key to encrypt a plaintext to secure data transmission. The difference to classical cryptographic techniques is that QKD relies on the quantum laws of physics. The key distributed between the communication parties is generated by single photon pulses. Classical cryptographic methods are based on unproven complexity assumptions whereas QKD is considered to be unconditionally secure and does not involve computational complexity. An unconditionally secure technique means that it is proven as an information theoretic secure method o secure data transmission. Hence an eavesdropper can listen to the communication but it will be detected by the communication partners because the eavesdropper induces noise on the channel.

Because of the fact that the security of a symmetric cryptosystem depends on the quality of the key, overall QKD solves the problem of secure key distribution between two adjacent nodes.

CHAPTER 4

Quantum Key Distribution

In this chapter, the basis for the quantum information theory as well as the basic concepts of quantum key distribution and why it is unconditional secure are described. We will start with a short introduction of classical information theory. However, the focus will be on the differences between bits and qubits, the necessary arithmetics and the basic model of quantum key distribution. This chapter describes the required basics for the experiments on quantum key distribution.

4.1 Classic Cryptography

Cryptography is the theory of secure communications between communication partners. Claude Elwood Shannon (* 1916 – † 2001) was the founder of modern cryptography. Through his work it was no longer necessary to keep cryptographic functions secret. He was the founder of secrecy systems in modern cryptography. In general, a deterministic encryption function is at least injective, since otherwise we can not conclude unambiguosly on the plaintext from the ciphertext. If an encryption function is not injective it is not deterministic, as well. The basic arithmetics and the definitions of functions can be found in Appendix A.

A function f is bijective if and only if f is injective and surjective. For every bijection there exists an inverse function f^{-1}. A secrecy system requires a bijection.

Let x denote the elements of the set of plaintexts and let y denote the elements of the set of ciphertexts. The encryption function is denoted by $f(x)$ and the decryption function by $f^{-1}(y)$.

$$y = f\left(x\right) \tag{4.1}$$

$$x = f^{-1}\left(y\right) \tag{4.2}$$

$$x = f^{-1}\left(y\right) = f^{-1}\left(f\left(x\right)\right) \tag{4.3}$$

To implement an instance of a secrecy system we introduce an arbitrary instance of a plaintext $(X = x)$, a ciphertext $(Y = y)$ and the secret keys (K_S) for the encryption and (K_P) for the decryption. Figure 4.1 shows a reversible, bijective function. Hence the encryption function and the decryption function are reversible, respectively. To encrypt the plaintext and decrypt

$$\begin{tikzcd} x_i \arrow[r, shift left, "E_{K_P}"] & y_j \arrow[l, shift left, "D_{K_S}"] \end{tikzcd}$$

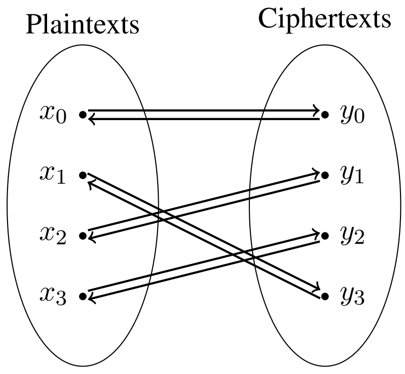


Figure 4.1: Secrecy Systems from Bijection and Reversibility - $f(x_i) = y_j$ and $f^{-1}(y_j) = x_i$

the ciphertext, an encryption function $(E_{K_P} = f)$ and a decryption function $\left(D_{K_S} = f^{-1}\right)$ are required. The notation $(E_{K_P} = f)$ means that the encryption function uses the key K_P to obtain the ciphertext from the plaintext and $\left(D_{K_S} = f^{-1}\right)$ means that the decryption function uses the key K_S to obtain the plaintext from the ciphertext. The encryption and decryption are defined as follows

$$Y = E_{K_P}(X), \tag{4.4}$$

and

$$X = D_{K_S}(Y). \tag{4.5}$$

Let A be the source (sender) and B be the destination (receiver). Public keys are denoted by K_P. They are assumed to be known by every possible communication partner i.e. even eavesdroppers know the public keys. If A wants to send a message X to B, it looks up the public key $K_{P,B}$ of B and uses Eq. (4.4) to obtain the ciphertext. The ciphertext Y is sent by A to B over the public channel. Let K_S be the private keys, which are kept secret. Hence B uses the decryption function $D_{K_{S,B}}$ as shown in Eq. (4.5) to obtain the plaintext from the ciphertext. This method allows to encrypt and decrypt messages for the two corresponding communication parties A and B. If A wants to sign the message to protect its integrity, we specify an encryption function

$$d = E_{K_S}(X) \tag{4.6}$$

and a decryption function

$$X = D_{K_P}(d). \tag{4.7}$$

X	Y	NOT(X)	NOT(Y)	AND	OR	NAND	NOR	XOR
0	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	0	1
1	0	0	1	0	1	1	0	1
1	1	0	0	1	1	0	0	0

Figure 4.2: Classic information - Logical bit - Overview

A encrypts the message with its private key $K_{S,A}$ and obtains $E_{K_{S,A}}(X) = d$. It generates the signature S_A and appends it after the original message (d, S_A). A encrypts the pair (d, S_A) with $B's$ public key $K_{P,B}$ and generates $E_{K_{P,B}}(d, S_A) = E_{K_{P,B}}\left(E_{K_{S,A}}(X), S_A\right) = e$. A sends the signed text e to B on the public channel. B uses its private key $K_{S,B}$ and computes $D_{K_{S,B}}(e) = D_{K_{S,B}}\left(E_{K_{P,B}}(d, S_A)\right) = (d, S_A)$. From the signature S_A B concludes the source A and uses the public key $K_{P,A}$ to decrypt the message and computes $D_{K_{P,A}}(d) = D_{K_{P,A}}\left(E_{K_{S,A}}(X)\right) = X$. Methods based on the procedures mentioned above, are called public key cryptography (PKC). In public key cryptography $K_S \neq K_P$ holds. PKC is based on number theoretic problems of unknown complexity (integer factorization, modular arithmetics, elliptic curves). That means its security is based on unproven complexity assumptions. If it is sufficiently difficult to calculate the inverse of a function, it is referred to as a one way function. Public key encryption and decryption requires a pair of keys, with one component beeing public and the other one kept secret. A major example for public key cryptography is the RSA (by Rivest, Shamir, Adleman) [42]. The public key is known to an arbitrary set of communication parties, i.e. eavesdroppers know the key, too. The private component is kept secret, in particular.

Symmetric key cryptography uses only secret keys and is formally known as secret key cryptography. It can use an identical key to encrypt and decrypt messages or as an alternative one can choose a pair of keys (K_S, K_P) where the keys can be calculated with a single operation from each other, respectively. In classical secret key cryptography the keys are identically, hence $K = K_S = K_P$ holds. The problem of secret key cryptography is the key exchange between the communication parties, because the key is assumed to kept secret. A cryptosystem can be information theoretic secure. That means it is secure even if an eavesdropper has unlimited time and computing power. Public key cryptography is secure because of its unproven computational hardness assumptions. An overview about basic cryptographic knowledge, public key cryptography and symmetric key cryptography and vocabulary is given in [25, 46]. In Fig. 4.2 an overview of classic bit operations is given. One can see that the logical XOR operation is reversible, but the AND, OR, NAND and NOR are not. The XOR operation can be interpreted as an addition modulo 2 i.e. XOR $(X, Y) = (X + Y) \bmod 2$. The bitwise XOR operation can be used to encrypt plaintexts and decrypt ciphertexts because of its reversibility. We want to encrypt plaintexts to protect the transmitted data against eavesdropping by an attacker who listens to the private communication between the authorized communication parties without their consent.

In 1949, Claude Elwood Shannon published a formal analysis of communication theory problems [43]. The work introduces communication theory for secrecy systems. To transmit an arbitrary plaintext from the source to the destination Shannon proposed the one time pad (OTP), a symmetric key encryption technique. The one time pad uses as key a random number of the same length as the plaintext to be transmitted over a public (probably unsecure) channel. The keys are used only once and kept secret. The OTP is an example for an information theoretic secury cryptosystem. However, it has two potential bottlenecks. Generating truly random keys in large quantities and distributing the key between all communication parties, so that Eve can not draw any conclusions on the secret key (formally known as key distribution problem) are the major problems. Quantum key distribution (QKD) offers a solution to this problem. This will be shown in further sections of this work. There are a variety of divergent key distribution techniques between Alice and Bob. A detailed explanation of classic key distribution methods is given in [1,46].

4.2 Quantum Information Theory

In quantum mechanics, quantum information is represented by the state of a quantum system. Quantum information theory is the theory of information processing within the laws of quantum mechanics which rests on four primary postulates.

Postulate 1: The state of a quantum mechanical system is specified by a wave function $\varphi\left(\hat{r},t\right)$. It describes the temporal and spatial evolution of a quantum mechanical particle and depends on the coordinates of the particle $(\hat{r}=(\hat{x},\hat{y},\hat{z}))$ and on time (t). The simplified wave function $\varphi\left(\hat{x},t\right)$ describes a particle with one degree of freedom of motion (x-axis). The state of the system is completely defined by a time-dependent state vector of a Hilbert space.

Postulate 2: Every dynamical observable corresponds to a linear, Hermitian operator in quantum mechanics. These operators act on the wave function $\varphi\left(\hat{x},t\right)$. That means that measurable properties of a system, e.g. the momentum $\hat{p}_x = i\hbar\frac{\partial}{\partial x}$, $\hat{p}_y = i\hbar\frac{\partial}{\partial y}$, $\hat{p}_z = i\hbar\frac{\partial}{\partial z}$, $\hat{p} = -i\hbar\nabla$, a position $\hat{r}$, or an energy $\hat{H} = \frac{\hat{p}^2}{2m} + V\left(\hat{x}\right)$ can be interpreted as observables. The reduced Planck constant (or Dirac constant) is denoted by $\hbar$. It is equal to the planck constant (h) divided by 2π i.e. $\hbar = \frac{h}{2\pi}$. $\hat{H}$ denotes the hamilton operator, where m is the mass of the particle and $V\left(\hat{x}\right)$ denotes the potential energy operator. The imaginary unit from complex numbers is denoted as $i^2 = -1$. The gradient operator is denoted by ∇. Hence observables are properties, that can be observed or measured.

Postulate 3: In quantum mechanics, one can only obtain eigenvalues by a measurement of observables. A Measurement of observables is a method to extract information form a state.

Let A be an arbitrary $n \times n$ matrix and v be a non-zero $n \times 1$ column vector. If v is multiplied with matrix A such that $\lambda v^{'} = Av$ holds, an eigenvector $v^{'}$ is a vector that is parallel to x and the eigenvalue corresponds to λ which can be interpreted as a scale factor to v.

Postulate 4: The temporal evolution of states in a system is determined by the Schrödinger equation

$$\hat{H}\varphi\left(\hat{x},t\right) = i\hbar\frac{\partial\varphi\left(\hat{x},t\right)}{\partial t}. \tag{4.8}$$

That means that a state vector evolves in time t. This chapter is inspired by [17,27,35,37,38].

Qubits

A quantum bit (qubit) is the smallest unit of quantum information such as the bit in classic information theory. The two orthonormal vectors $|0\rangle$ and $|1\rangle$ are the basis states. Qubits are denoted by the Bra-Ket notation to describe a state of a quantum system. It was introduced by Paul Adrien Maurice Dirac (* 08.08.1902 – † 20.10.1984). The $n \times 1$ column vector (ket-vector) is denoted by

$$|\psi\rangle = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} \tag{4.9}$$

and the corresponding $1 \times n$ row vector (bra-vector) is denoted by

$$\langle\psi| = (a_1^*, a_2^* \dots a_n^*) \,. \tag{4.10}$$

The bra-vector can be built by the transpose $(*)$ of the ket-vector. A qubit $|\psi\rangle$ is an arbitrary unit vector in the two-dimensional, complex vector space $\mathbb{C}^2$ spanned by $|0\rangle$ and $|1\rangle$ over the complex numbers, which denotes the main conceptual difference between a bit and a qubit. The general state of a qubit can be specified as follows

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle, \tag{4.11}$$

where $\alpha, \beta \in \mathbb{C}$ are complex numbers. The only restriction for $|\psi\rangle$ is the normalization condition, i.e. the length of the unit vector has to be 1. The state $|\psi\rangle$ is called a superposition of the states $|0\rangle$ and $|1\rangle$ with the amplitudes $\alpha, \beta \in \mathbb{C}$ s.t.

$$|\alpha|^2 + |\beta|^2 = 1. \tag{4.12}$$

A function mapping from an arbitrary qubit to a classic bit value results in "0" / "1" with probability $p\,(0)\,,\ p\,(1)$. A generalization for n qubits is given below

$$|\psi\rangle = \sum_{0 \le x < 2^n} \alpha_x\,|x\rangle_n, \tag{4.13}$$

$$\sum_{0 \le x < 2^n} |\alpha_x|^2 = 1. \tag{4.14}$$

The set of 2^n classical states is called computational basis. From Eq.(4.12) follows that an arbitrary, single qubit state can be denoted as a Bloch vector $|\psi\rangle$ as shown below.

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}\,|1\rangle \tag{4.15}$$

where $\theta, \varphi \in \mathbb{R}$ and represent angles in the Bloch sphere by the Swiss physicist Felix Bloch (* 23.10.1905 – † 10.09.1983) as shown in Fig. 4.3. A geometric interpretation of qubit states, as coordinates on the surface of a unit sphere, is given by the Bloch sphere. It generalizes a complex number z s.t. $|z|^2 = 1$ is denoted by a point on the unit sphere in polar coordinates.

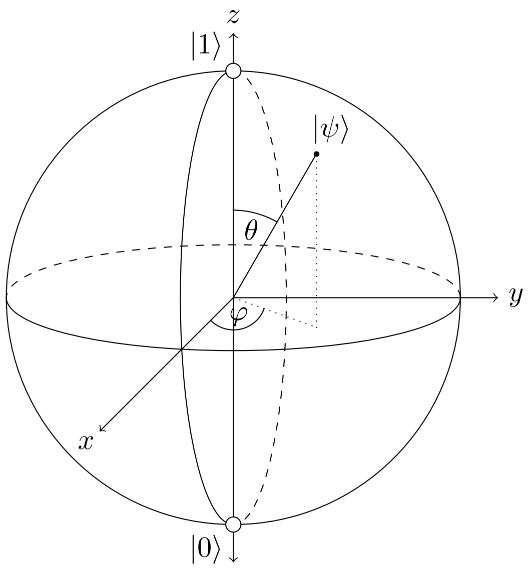


Figure 4.3: Geometrical Representation of the correspondence of the angles θ, φ to the Bloch State $|\psi\rangle$ - $0 \leq \theta \leq \pi,\ 0 \leq \varphi \leq 2\pi$

Let $e^{i\varphi}\ |\psi\rangle$ be an arbitrary state, where $|\psi\rangle$ denotes the state vector and $\varphi \in \mathbb{R}$ is a real number. Hence the states $e^{i\varphi}\ |\psi\rangle$ and $|\psi\rangle$ are equal up to a so called global phase $e^{i\varphi}$. Hence both amplitudes differ by the factor $e^{i\varphi}$.The computational basis is denoted by a column vector as shown in Eq.(4.9). The most common representation for the $\mathbb{C}^2$ is the interpretation of ψ $(I\ (\psi))$ as follows

$$|\psi\rangle = \begin{cases} \begin{pmatrix} 1 \\ 0 \end{pmatrix}, & I(\psi) = 0 \\ \begin{pmatrix} 0 \\ 1 \end{pmatrix}, & I(\psi) = 1 \end{cases} \tag{4.16}$$

Hence $|0\rangle$ and $|1\rangle$ form the computational basis with the according ket-vector as shown in Eq.(4.16). The corresponding bra-vector is shown below.

$$\langle\psi| = \begin{cases} \begin{pmatrix} 1, 0 \end{pmatrix}, & I(\psi) = 0 \\ \begin{pmatrix} 0, 1 \end{pmatrix}, & I(\psi) = 1 \end{cases} \tag{4.17}$$

Consider the states $|+\rangle$ and $|-\rangle$ s.t.

$$|\psi\rangle = \begin{cases} \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), & I\ (\psi) = + \\ \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right), & I\ (\psi) = - \end{cases} \tag{4.18}$$

By contrast to the global phase, these states differ in their amplitudes. Hence, the amplitude of state $|+\rangle$ is $\frac{1}{\sqrt{2}}$, whereas it is $(-1)\ \frac{1}{\sqrt{2}}$ in the complementary state $|-\rangle$, respectively. Two amplitudes of some states α_i, α_j, differ by a relative phase if $\alpha_i = e^{i\varphi}\alpha_j$ holds for some $\varphi \in \mathbb{R}$ and $i \neq j$.

Linear Operators

A single state will be mapped linearly to another state via linear operators (i.e. functions, linear transformations) and will be called a state change. Operations on a single bit value in classic computation are performed by the gate operations mentioned in Fig. 4.2. In quantum computing, a qubit can exist in a superposition. Hence it is a little bit more complex than the logical operations from Fig. 4.2. Mathematical functions are used to describe operations on qubits, to change the state from one into another.

Let $n, m \in \mathbb{N}$ be arbitrary integers. Let $A : \mathcal{V} \to \mathcal{W}$ be a linear operator and $|v_j\rangle$ be an orthonormal basis of $\mathcal{V}$ and $|w_i\rangle$ an orthonormal basis of $\mathcal{W}$ s.t. $1 \leq j \leq n$ and $1 \leq i \leq m$ holds. The linear operator represents a transition from vector space $\mathcal{V}$ to $\mathcal{W}$ where $\mathcal{V}, \mathcal{W} \in \mathbb{C}^*$. Let $A(n,m)$ be a $n \times m$ matrix, that fulfills the linearity equation and let $|v\rangle \in \mathbb{C}^m$ be a vertical bar vector. There exist $A_{nj} \in \mathbb{C}$ s.t.

$$A\left(\sum_i a_i |v_i\rangle\right) = \sum_i \left(a_i A |v_i\rangle\right) \tag{4.19}$$

and

$$A|v_j\rangle = \sum_i \left(A_{ij} |w_i\rangle\right) \tag{4.20}$$

holds. We get the matrix representation of a ket-vector from Eq.(4.19) and Eq.(4.20). The computational basis can be denoted as follows.

$$|v\rangle = a_1|0\rangle + a_2|1\rangle + \cdots + a_n|n\rangle = \sum_{1 \leq i,j \leq n} a_j |v_j\rangle = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \tag{4.21}$$

$$|w\rangle = a_1|0\rangle + a_2|1\rangle + \cdots + a_n|n\rangle = \sum_{1 \leq i,j \leq n} a_j |v_j\rangle = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \tag{4.22}$$

i.e. there exists only one value in the column vector that is mapped to 1.

$$I(a_i) = \begin{cases} 0, & i = j \\ 1, & i \neq j \end{cases} \tag{4.23}$$

Let A be a 2×2 matrix s.t. $a, b, c, d, e, f \in \mathbb{R}$ i.e.

$$A = \begin{pmatrix} a + ib & c - id \\ e & if \end{pmatrix} \tag{4.24}$$

A matrix A^* is the complex conjugate matrix of matrix A, if and only if $\mathfrak{Re}(z^*) = \mathfrak{Re}(z)$ and $\mathfrak{Im}(z^*) = -\mathfrak{Im}(z)$ holds. Furthermore, a matrix $\left(A^T\right)_{n\times m}$ is the transpose of matrix $A_{m\times n}$

if and only if $\left(A^T\right)_{n\times m}$ is the reflection over matrix $A's$ main diagonal. Hence, an interchange of columns and rows succeeds over $n \times m.$ From the sequential composition by the complex conjugate A^* and the transposition A^T via $(A^*)^T$ follows the matrix A^*

$$A^* = \begin{pmatrix} a-ib & e \\ c+id & -if \end{pmatrix} \tag{4.25}$$

and

$$(A^*)^T = A^H = \begin{pmatrix} a-ib & c+id \\ e & -if \end{pmatrix} \tag{4.26}$$

Let $z = a + ib \in \mathbb{C}$ s.t. $a, b \in \mathbb{R}$ and let A be a linear transformation. Furthermore let $|\psi\rangle$ be an arbitrary orthonormal vector, so we apply A on $|\psi\rangle$. Due to linearity follows

$$A\left(a\,|\psi\rangle + b\,|\phi\rangle\right) = aA\,|\psi\rangle + bA\,|\phi\rangle \tag{4.27}$$

and if it is not explicitly written the following notation holds.

$$|A\psi\rangle = A\,|\psi\rangle \tag{4.28}$$

From a linear operation A, follows another operation.

$$\langle Av| = \langle v|\,A^\dagger \tag{4.29}$$

The new transformation is called the adjoint operator ($\dagger$), e.g. on the dual space of bra vectors. From that follows the adjoint of the $|v\rangle$ vector

$$(|v\rangle)^\dagger = \langle v|. \tag{4.30}$$

Let A be a binary operation and $\dagger$ be the adjoint operator. An operator is hermitian if and only if $A^\dagger = A$ holds.

$$(|v\rangle, A|w\rangle) = \left(A^\dagger|v\rangle, |w\rangle\right) \tag{4.31}$$

Inner Products

Inner Products (or Scalar products) of two vectors are transformations from a vector space, to a complex number. An inner product $|v\rangle, |w\rangle$, where $|v\rangle$ is associated with $\langle v|$ will be represented in braket notation as follows

$$\langle v|\,(|w\rangle) = \langle v|w\rangle = \sum_{1\leq i\leq n} a_i^* b_i = (a_1^* \dots a_n^*)\begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \tag{4.32}$$

The inner product satisfies the following requirements.

$$\left(|v\rangle, \sum_{1\leq i\leq n} \lambda_i\,|w_i\rangle\right) = \sum_{1\leq i\leq n} \lambda_i\,(|v\rangle, |w_i\rangle) \tag{4.33}$$

$$(|v\rangle, |w\rangle) = (|v\rangle, |w\rangle)^* \tag{4.34}$$

$$(|v\rangle, |v\rangle) \geq 0 \text{ iff } |v\rangle = 0 \tag{4.35}$$

Scalar products are linear in the second (right) vector argument. An inner product of two vectors in a vector space over $\mathbb{C}^*$ is denoted as follows

$$\langle v|w\rangle = \langle v|w\rangle^* \tag{4.36}$$

s.t.

$$(a + ib)^* = (a - ib) \text{ where } a, b \in \mathbb{R} \tag{4.37}$$

and from that definition follows the inner product of itself (complex conjugate) is defined as follows:

$$\langle v|v\rangle = \sum_{1 \leq i \leq n} |a_i|^2 \, s.t. \tag{4.38}$$

$$|a + ib|^2 = a^2 + b^2 \text{ where } a, b \in \mathbb{R} \tag{4.39}$$

Furthermore the inner product of a vector with itself is a real number s.t.

$$\langle v|v\rangle > 0, |v\rangle \neq 0 \tag{4.40}$$

Outer Products

By contrast to scalar products, there exists the outer product. A matrix $A_{n,m}$ is used instead of vectors where $|v\rangle \in \mathcal{V}, |w\rangle \in \mathcal{W}, (|v\rangle \times |w\rangle) \in (\mathcal{V} \times \mathcal{W})$ holds, but $|v\rangle, |w\rangle \notin (\mathcal{V} \times \mathcal{W})$ holds. While the inner product results in a scalar $\mathbb{C}$, the result by applying an outer product on a pair of vectors, is a matrix $A_{i,j}$.

$$|v\rangle\langle w| = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} (b_1^* \ldots b_m^*) = \begin{pmatrix} a_1 b_1^* & \ldots & a_1 b_m^* \\ \vdots & \ddots & \vdots \\ a_n b_1^* & \ldots & a_n b_m^* \end{pmatrix} = A_{i,j} \tag{4.41}$$

Furthermore the unit operator 1 follows for a complete orthonormal set

$$\sum_{1 \leq i \leq n} |i\rangle\langle i| = 1 \tag{4.42}$$

From the unity equation follows

$$|v\rangle = 1\,|v\rangle = \sum |i\rangle\langle i|v\rangle \tag{4.43}$$

Tensor Product

A Tensor product is denoted by $\otimes$. It differs from the inner and outer product. However, the tensor product results in a vector space, from two vector spaces.

Let A, B be linear operations $A : \mathcal{V} \to \mathcal{V}^{'}$ and $B : \mathcal{W} \to \mathcal{W}^{'}$. Furthermore let $A \otimes B : \mathcal{V} \otimes \mathcal{W} \to \mathcal{V}^{'} \otimes \mathcal{W}^{'}$. Suppose $A_{m,n}, B_{o,p}$ be matrices, s.t. $m, n, o, p \in \mathbb{N}$ we get

$$\begin{aligned}
A \otimes B &= \begin{bmatrix} A_{11}B & A_{12}B & \cdots & A_{1n}B \\ A_{21}B & A_{22}B & \cdots & A_{2n}B \\ \cdots & \cdots & \cdots & \cdots \\ A_{m1}B & A_{m2}B & \cdots & A_{mn}B \end{bmatrix} \\
&= \begin{bmatrix}
a_{11}b_{11} & a_{11}b_{12} & \cdots & a_{11}b_{1p} & \cdots & \cdots & a_{1n}b_{11} & a_{1n}b_{12} & \cdots & a_{1n}b_{1p} \\
a_{11}b_{21} & a_{11}b_{22} & \cdots & a_{11}b_{2p} & \cdots & \cdots & a_{1n}b_{21} & a_{1n}b_{22} & \cdots & a_{1n}b_{2p} \\
\vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\
a_{11}b_{o1} & a_{11}b_{o2} & \cdots & a_{11}b_{op} & \cdots & \cdots & a_{1n}b_{o1} & a_{1n}b_{o2} & \cdots & a_{1n}b_{op} \\
\vdots & \vdots & \ddots & \vdots & \ddots & & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \ddots & \vdots & & \ddots & \vdots & \vdots & \ddots & \vdots \\
a_{m1}b_{11} & a_{m1}b_{12} & \cdots & a_{m1}b_{1p} & \cdots & \cdots & a_{mn}b_{11} & a_{mn}b_{12} & \cdots & a_{mn}b_{1p} \\
a_{m1}b_{21} & a_{m1}b_{22} & \cdots & a_{m1}b_{2p} & \cdots & \cdots & a_{mn}b_{21} & a_{mn}b_{22} & \cdots & a_{mn}b_{2p} \\
\vdots & \vdots & \ddots & \vdots & & \ddots & \vdots & \vdots & \ddots & \vdots \\
a_{m1}b_{o1} & a_{m1}b_{o2} & \cdots & a_{m1}b_{op} & \cdots & \cdots & a_{mn}b_{o1} & a_{mn}b_{o2} & \cdots & a_{mn}b_{op}
\end{bmatrix}
\end{aligned} \tag{4.44}$$

Let $\mathcal{V}, \mathcal{W}$ be m, n dimensional vector spaces. From $\mathcal{V} \otimes \mathcal{W}$ and $|v\rangle \in \mathcal{V}, |w\rangle \in \mathcal{W}$ follows the m, n dimensional vector space. Furthermore let $|v\rangle = \alpha_v |0\rangle + \beta_v |1\rangle$ and $|w\rangle = \alpha_w |0\rangle + \beta_w |1\rangle$ be two states of qubits. Let the state $|\Psi\rangle = |v\rangle \otimes |w\rangle$ be a new state, constructed by the tensor product.

$$\begin{aligned}
|\Psi\rangle &= |v\rangle \otimes |w\rangle \\
&= (\alpha_v |0\rangle + \beta_v |1\rangle) \otimes (\alpha_w |0\rangle + \beta_w |1\rangle) \\
&= \alpha_v\alpha_w |00\rangle + \alpha_v\beta_w |01\rangle + \beta_v\alpha_w |10\rangle\beta_v\beta_w |11\rangle \\
&= \begin{pmatrix} \alpha_v\alpha_w \\ \alpha_v\beta_w \\ \beta_v\alpha_w \\ \beta_v\beta_w \end{pmatrix}
\end{aligned} \tag{4.45}$$

To construct an example for the tensor product, the main components are interpreted as follows

$$I(\alpha_v) = 1,\ I(\beta_v) = 2,\ I(\alpha_w) = 3,\ I(\beta_w) = 4$$

$$|\Psi\rangle = |v\rangle \otimes |w\rangle = (\alpha_v |0\rangle + \beta_v |1\rangle) \otimes (\alpha_w |0\rangle + \beta_w |1\rangle) = \begin{pmatrix} 1\cdot 3 \\ 1\cdot 4 \\ 2\cdot 3 \\ 2\cdot 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 6 \\ 8 \end{pmatrix} \tag{4.46}$$

The result of the tensor product is the transponse of the row vector $\langle\Psi| = (3,\ 4,\ 6,\ 8)$.

Hadamard Gate

The Hadamard gate is a linear operator and is denoted by H. It is defined by the following matrix

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{4.47}$$

The hadamard operator produces an equally weighted superposition of $|0\rangle$ and $|1\rangle$. In the next steps we apply the Hadamard Operation on the states (ket vectors) $|0\rangle, |1\rangle$ as follows

$$\begin{aligned} H\,|0\rangle &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \left[\begin{pmatrix} 1\cdot 1 + 1\cdot 0 \\ 1\cdot 1 + (-1)\cdot 0 \end{pmatrix} \right] \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \left[\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] \\ &= \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \end{aligned} \tag{4.48}$$

$$\begin{aligned} H\,|1\rangle &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \left[\begin{pmatrix} 1\cdot 0 + 1\cdot 1 \\ 1\cdot 0 + (-1)\cdot 1 \end{pmatrix} \right] \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \left[\begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] \\ &= \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \end{aligned} \tag{4.49}$$

Remember the states from Eq. (4.48) and Eq. (4.49). We apply the hadamard transformation on the states as follows.

$$\begin{aligned}
H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) &= \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&= \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle + \frac{1}{2}|0\rangle - \frac{1}{2}|1\rangle \\
&= |0\rangle
\end{aligned} \tag{4.50}$$

$$\begin{aligned}
H\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) &= \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) - \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&= \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle - \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle \\
&= |1\rangle
\end{aligned} \tag{4.51}$$

Two-dimensional NOT Gate

An operation like erase is an irreversible operation, so far. Quantum Computers are calculating by reversible operations, where an initial state is transformed to a final state, using reversible operations. If a state is reversible, there exists exactly one preimage, as shown in figure 4.1. A reversible transformation is the NOT operation (bitflip) and will be denoted as X. This operation interchanges the values between a pair of states s.t.

$$X : |\psi\rangle \rightarrow |\psi'\rangle, \text{ s.t. } 1' = 0,\ 0' = 1 \tag{4.52}$$

A sequential composition of the bitflip operation is defined as follows

$$X \circ X = X \cdot X = X^2 = I \tag{4.53}$$

Hence the matrix representation of the NOT operator is given as follows

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{4.54}$$

on a 2-dimensional vector space.

Two-dimensional Unit Gate

From the sequential composition of the two-dimensional NOT gates X^2 follows the two-dimensional unit gate. It is denoted by

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{4.55}$$

Two-dimensional Pauli Gates

The pauli gates are linear operators (single qubit quantum gates) and are denoted by σ_X, σ_Y and σ_Z. Hence, these linear operators proceed on one single qubit. The following equations show the matrix representations of the 2×2 three basic pauli matrices.

$$\sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{4.56}$$

$$\sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{4.57}$$

$$\sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{4.58}$$

By performing a matrix multiplication with σ_Z and σ_X and the fact, that $i^2 = (-1)$ follows the modified Pauli gate $i\sigma_Y$.

$$\sigma_Z \sigma_X = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = i \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = i\sigma_Y \tag{4.59}$$

$$i\sigma_Y = i \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{4.60}$$

From the Pauli gates follows the computation for each instance of σ.

$$\sigma_X \sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 \cdot 0 + 1 \cdot 1 & 0 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 0 + 0 \cdot 1 & 1 \cdot 1 + 0 \cdot 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.61}$$

$$\sigma_Y \sigma_Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = \begin{pmatrix} 0 \cdot 0 + (-i) \cdot i & 0 \cdot (-i) + (-i) \cdot 0 \\ i \cdot 0 + i \cdot 0 & i \cdot (-i) + 0 \cdot 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.62}$$

$$\sigma_Z \sigma_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 \cdot 1 + 0 \cdot 0 & 1 \cdot 0 + 0 \cdot (-1) \\ 0 \cdot 1 + (-1) \cdot 0 & 0 \cdot 0 + (-1) \cdot (-1) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.63}$$

Because the pauli matrices are self adjoint, a sequential composition of σ results in the 2-dimensional unit matrix.

$$\sigma_X^2 = \sigma_Y^2 = \sigma_Z^2 = I \tag{4.64}$$

In the next steps we apply the Pauli gate operations on the ket vectors, as follows.

$$\sigma_X \left|0\right\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \left[\begin{pmatrix} 0 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 1 + 0 \cdot 0 \end{pmatrix} \right] = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{4.65}$$

$$\sigma_X \left|1\right\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \left[\begin{pmatrix} 0 \cdot 0 + 1 \cdot 1 \\ 1 \cdot 0 + 0 \cdot 1 \end{pmatrix} \right] = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{4.66}$$

A σ_X operation on the states results in a bitflip. That means that the state will be changed from $|0\rangle$ to $|1\rangle$ and vice versa.

$$\sigma_Z\,|0\rangle = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \left[\begin{pmatrix}1\cdot 1 + 0\cdot 0\\ 0\cdot 1 + (-1)\cdot 0\end{pmatrix}\right] = \begin{pmatrix}1\\0\end{pmatrix} \tag{4.67}$$

$$\sigma_Z\,|1\rangle = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \left[\begin{pmatrix}1\cdot 0 + 0\cdot 1\\ 0\cdot 0 + 1\cdot(-1)\end{pmatrix}\right] = \begin{pmatrix}0\\-1\end{pmatrix} \tag{4.68}$$

$$\sigma_Z\,|+\rangle = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{pmatrix} = \left[\begin{pmatrix}1\cdot\frac{1}{\sqrt{2}} + 0\cdot\frac{1}{\sqrt{2}}\\ 0\cdot\frac{1}{\sqrt{2}} + (-1)\cdot\frac{1}{\sqrt{2}}\end{pmatrix}\right] = \begin{pmatrix}\frac{1}{\sqrt{2}}\\ -\frac{1}{\sqrt{2}}\end{pmatrix} \tag{4.69}$$

$$\sigma_Z\,|-\rangle = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\ -\frac{1}{\sqrt{2}}\end{pmatrix} = \left[\begin{pmatrix}1\cdot\frac{1}{\sqrt{2}} + 0\cdot\frac{1}{\sqrt{2}}\\ 0\cdot\frac{1}{\sqrt{2}} + (-1)\cdot(-1)\cdot\frac{1}{\sqrt{2}}\end{pmatrix}\right] = \begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{pmatrix} \tag{4.70}$$

The $i\sigma_Y$ operator is the sequential composition of the operators σ_X and σ_Z. As a result, the operator $i\sigma_Y$ performs both flip operations, the bitflip and the phase flip.

$$\begin{aligned} i\sigma_Y\,|0\rangle &= i\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\\ &= \begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\\ &= \left[\begin{pmatrix}0\cdot 1 + 1\cdot 0\\ (-1)\cdot 1 + 0\cdot 0\end{pmatrix}\right]\\ &= \begin{pmatrix}0\\-1\end{pmatrix}\end{aligned} \tag{4.71}$$

$$\begin{aligned} i\sigma_Y\,|1\rangle &= i\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\\ &= \begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\\ &= \left[\begin{pmatrix}0\cdot 0 + 1\cdot 1\\ (-1)\cdot 0 + 0\cdot 1\end{pmatrix}\right]\\ &= \begin{pmatrix}1\\0\end{pmatrix}\end{aligned} \tag{4.72}$$

A summarization of the Pauli gate operations is given in Fig. 4.4.

Rotation Gate

A rotation gate transformation is a general purpose class of operators. Let θ be an arbitrary angle. A rotation operator rotates the plane by θ. Hence the rotation gate can be described by a

Gate	Qubit	Result	Operation
σ_X	$\|0\rangle$	$\begin{pmatrix}0\\1\end{pmatrix}$	Bit Flip
	$\|1\rangle$	$\begin{pmatrix}1\\0\end{pmatrix}$	
σ_Z	$\|0\rangle$	$\begin{pmatrix}1\\0\end{pmatrix}$	Phase Flip
	$\|1\rangle$	$\begin{pmatrix}0\\-1\end{pmatrix}$	
	$\|+\rangle$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\-\frac{1}{\sqrt{2}}\end{pmatrix}$	
	$\|-\rangle$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$	
$i\sigma_Y$	$\|0\rangle$	$\begin{pmatrix}0\\-1\end{pmatrix}$	Bit + Phase Flip
	$\|1\rangle$	$\begin{pmatrix}1\\0\end{pmatrix}$	

Figure 4.4: Pauli Gate Operations - Overview

2×2 rotation matrix. The rotation matrix is defined as an orthogonal matrix with determinant +1 and is shown in the following equation.

$$R_\theta = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}. \tag{4.73}$$

The determinant $detR_\theta$ is given below

$$detR_\theta = \begin{vmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{vmatrix} = \cos\theta \cdot \cos\theta - \sin\theta \cdot (-1) \cdot \sin\theta = 1. \tag{4.74}$$

As an alternative, one can choose the (2×2) rotation matrix with the same properties as shown above.

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{4.75}$$

The commutativity of the rotation gate follows immediately from the matrix multiplication and the addition theorems of trigonometric functions.

Let $\theta, \phi \in \mathbb{R}$ be arbitrary angles and R be the (2×2) rotation matrix, respectively.

$$\begin{aligned}
R_\theta \cdot R_\phi &= \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \\
&= \begin{pmatrix} \cos\theta \cdot \cos\phi + \sin\theta \cdot (-1) \cdot \sin\phi & \cos\theta \cdot \sin\phi + \sin\theta \cdot \cos\phi \\ (-1) \cdot \sin\theta \cdot \cos\phi + \cos\theta \cdot (-1) \cdot \sin\phi & (-1) \cdot \sin\theta \cdot \sin\phi + \cos\theta \cdot \cos\phi \end{pmatrix} \\
&= \begin{pmatrix} \cos(\theta+\phi) & \sin(\theta+\phi) \\ -\sin(\theta+\phi) & \cos(\theta+\phi) \end{pmatrix}
\end{aligned} \tag{4.76}$$

Let $|\psi\rangle = \alpha R_\theta |0\rangle + \beta R_\theta |1\rangle$ be an arbitrary state. By applying the rotation operator on $|\psi\rangle$ one can see, that the equation

$$\begin{aligned}
R_\theta |\psi\rangle &= \alpha R_\theta |0\rangle + \beta R_\theta |1\rangle \\
&= \alpha \left[\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right] + \beta \left[\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] \\
&= \alpha \left[\begin{pmatrix} 1 \cdot \cos\theta + 0 \cdot \sin\theta \\ 1 \cdot (-\sin\theta) + 0 \cdot \cos\theta \end{pmatrix} \right] + \beta \left[\begin{pmatrix} 0 \cdot \cos\theta + 1 \cdot \sin\theta \\ 0 \cdot (-\sin\theta) + 1 \cdot \cos\theta \end{pmatrix} \right] \\
&= \alpha \begin{pmatrix} \cos\theta \\ -\sin\theta \end{pmatrix} + \beta \begin{pmatrix} \sin\theta \\ \cos\theta \end{pmatrix} \\
&= \alpha \left[\begin{pmatrix} \cos\theta \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ -\sin\theta \end{pmatrix} \right] + \beta \left[\begin{pmatrix} \sin\theta \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \cos\theta \end{pmatrix} \right] \\
&= \alpha \left[\cos\theta \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (-\sin\theta) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] + \beta \left[\sin\theta \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \cos\theta \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] \\
&= \alpha \left[\cos\theta |0\rangle - \sin\theta |1\rangle \right] + \beta \left[\sin\theta |0\rangle + \cos\theta |1\rangle \right] \\
&= \alpha \cos\theta |0\rangle - \alpha \sin\theta |1\rangle + \beta \sin\theta |0\rangle + \beta \cos\theta |1\rangle \\
&= (\alpha \cos\theta + \beta \sin\theta) \; |0\rangle - (\alpha \sin\theta - \beta \cos\theta) \; |1\rangle \\
&= \alpha' \; |0\rangle + (-1) \, \beta' \; |1\rangle
\end{aligned} \tag{4.77}$$

results in a modified amplitude α', β' s.t. $\alpha' = (\alpha \cos\theta + \beta \sin\theta)$ and $\beta' = -\,(\alpha \sin\theta - \beta \cos\theta)$ and a change of the relative phase from $(+1)$ to (-1) of the state of the qubit.

CNOT Gate

The controlled not gate, also known as $CNOT$ gate is an essential operator and denoted by $CNOT_{ij}$. It manipulates 2 qubits i and j. However, both qubits are to be distinguished in once the control and secondly the target qubit. If the state of the i^{th} qubit is mapped to $|0\rangle$, the state of the j^{th} qubit is invariant, whereas if the state of the i^{th} qubit is mapped to $|1\rangle$, the $CNOT$ operator changes the value of the j^{th} qubit. It does this by applying a $\bmod 2$ addition. This is shown in the subscript $CNOT_{ij}$, where the first index denotes the control (source) qubit and the second index denotes the target qubit. The control qubit is invariant, whereas the target qubit changes if and only if the control qubit is $|1\rangle$. If the control qubit is on the left, the $CNOT$ operation leaves the states $|00\rangle$ and $|01\rangle$ invariant. However, the states $|10\rangle$ and $|11\rangle$ are exchanged. The matrix representation of $CNOT_{10}$ is given below.

$$CNOT_{10} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.78}$$

The CNOT gate can alternatively be given as $CNOT_{01}$, where the subscript means that the right qubit is the control, whereas the left one, is the target qubit. Hence, the states $|00\rangle$ and $|10\rangle$ are invariant, whereas the states $|01\rangle$ and $|11\rangle$ are exchanged. From this consideration the matrix representation of $CNOT_{01}$ is as follows.

$$CNOT_{01} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{4.79}$$

Furthermore, if the $CNOT$ operator is multiplied with itself, we obtain the identity matrix I immediately. Hence the equation below holds for $CNOT_{10}$

$$CNOT_{10}^2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I \tag{4.80}$$

whereas

$$CNOT_{01}^2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I \tag{4.81}$$

holds for $CNOT_{01}$. Because I denotes the 4×4 identity matrix and $AA^{-1} = A^{-1}A = I$ holds, we obtain that $CNOT_{ij} = CNOT_{ij}^{-1}$ holds.

Furthermore, two different cases need to be distinguished by the application of the $CNOT$ transformation on two qubits. The first is the application on pure 2-qubit states as follows a 2^2 variation with repition for $CNOT_{10}$

$$\begin{array}{ll} CNOT_{10}\,|00\rangle \rightarrow |00\rangle & CNOT_{10}\,|10\rangle \rightarrow |11\rangle \\ CNOT_{10}\,|01\rangle \rightarrow |01\rangle & CNOT_{10}\,|11\rangle \rightarrow |10\rangle \end{array} \tag{4.82}$$

and for the $CNOT_{01}$ transformation

$$\begin{array}{ll} CNOT_{01}\,|00\rangle \rightarrow |00\rangle & CNOT_{10}\,|10\rangle \rightarrow |10\rangle \\ CNOT_{01}\,|01\rangle \rightarrow |11\rangle & CNOT_{10}\,|11\rangle \rightarrow |01\rangle \end{array} \tag{4.83}$$

The second application is to apply the $CNOT$ transformation on a control qubit in a superposition of the one-qubit states $|0\rangle$ and $|1\rangle$. The results of the transformations are defined as $CNOT_{ij}\,|\psi\rangle\,|\varphi\rangle$.

Let $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ be an arbitrary superposition of the two states $|0\rangle$ and $|1\rangle$. Furthermore let the target qubit state be defined by $|\varphi\rangle \in \{|0\rangle, |1\rangle\}$. The results of the operation $CNOT_{10}$ are shown in the equations below.

$$\begin{aligned} CNOT_{10}\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\ |0\rangle = CNOT_{10}\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ CNOT_{10}\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\ |1\rangle = CNOT_{10}\frac{1}{\sqrt{2}}(|01\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \end{aligned} \tag{4.84}$$

As an alternative, let $|\varphi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ be an arbitrary superposition of the two states $|0\rangle$ and $|1\rangle$ and $\psi \in \{|0\rangle, |1\rangle\}$ be the target qubit. The $CNOT_{01}$ operation results are as follows.

$$\begin{aligned} CNOT_{01}\ |0\rangle\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = CNOT_{01}\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ CNOT_{01}\ |1\rangle\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = CNOT_{01}\frac{1}{\sqrt{2}}(|10\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|10\rangle + |01\rangle) \end{aligned} \tag{4.85}$$

Hence the $CNOT$ gate is similar to the reversible XOR gate from classical information theory. A compact definition is given below.

$$CNOT_{10}\ |\psi\rangle\ |\varphi\rangle = |\psi\rangle\ |\varphi \oplus \psi\rangle \quad CNOT_{01}\ |\psi\rangle\ |\varphi\rangle = |\psi \oplus \varphi\rangle\ |\varphi\rangle \tag{4.86}$$

Universal Quantum Gates

In classical computation, the set of all possible gates can be simulated by the $NAND$ operation as shown in Fig. 4.2. However, in the field of quantum information theory, exists at least one gate, that cannot be simulated by the minimal set of universal quantum gates. However, every quantum gate can approximately be simulated with a sufficient accuracy. Hence a minimal set of universal quantum gates can be found in [38]. This set contains the $CNOT$ gate, the hadamard gate, the phase gate S and the $\frac{\pi}{8}$-gate.

$$CNOT_{10} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.87}$$

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{4.88}$$

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{4.89}$$

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{pmatrix} \tag{4.90}$$

The proof for their universality can be found in [38], too.

Measurement of a qubit

By contrast to reversible operations, the only not reversible operation in the field of quantum information is the measurement. A measurement results in an interpretation of information, after a qubit has changed to its final state. Measured by linear operators, a measurement is defined by a transformation. We call it measurement operator, or hermitian operator M_m. Hence a system is in state $|\psi\rangle$ before the measurement occurs and in state $|\psi^{'}\rangle$ after the measurement. Furthermore, the results of a measurement will be part of classical information theory. Hence the results are classic bit values. The probability of a measurement of the qubit in state $|\psi\rangle$, that m occurs and $m \in \{0, 1\}$ is denoted by $p\,(m)$ and defined as follows.

$$p(m) = \langle\varphi|M_m^{\dagger}\ M_m|\varphi\rangle \tag{4.91}$$

Hence, the measurement operator M_m is applied on the state before the measurement $|\psi\rangle$. Furthermore, the adjoint $M_m^{\dagger}$ is applied on the bra $\langle\psi|$.

$$|\psi^{'}\rangle = U\ |\psi\rangle \tag{4.92}$$

The state after the measurement can be described by the following equation.

$$|\psi^{'}\rangle = \frac{M_m|\psi\rangle}{\sqrt{\langle\psi|M_m^{\dagger}M_m|\psi\rangle}} \tag{4.93}$$

For the set of all measurement operators, the completeness equation holds.

$$\sum_m M_m^{\dagger}M_m = I \tag{4.94}$$

Hence, from completeness follows that the sum of all probabilities for all measurement operators is 1.

$$\sum_m p(m) = 1 = \sum_m \langle\psi|M_m^{\dagger}\ M_m|\psi\rangle = 1 \tag{4.95}$$

A set of measurements exists. The most important are the measurements in the computational basis. As shown above, the states $|0\rangle, |1\rangle$ form a computational basis for $\mathbb{C}^2$.

Let $|0\rangle, |1\rangle$ be two computational bases. The measurement in the computational basis is defined as the outer product. Hence it is a mapping to the classic states 0 and 1, from classic information theory.

Let M_m be a measurement operator s.t. $m \in \{0, 1\}$.

$$M_0 = |0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad M_1 = |1\rangle\langle 1| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.96}$$

From the definition of M_m follows immediately, that $M_m = M_m^{\dagger}$ holds. Now we apply the measurement operator on an arbitrary state of a qubit $|\psi\rangle = \alpha\ |0\rangle + \beta\ |1\rangle$, to conclude the probability of the result of the measurement, that $m = 0$.

$$p\,(m) = \langle\psi|M_m^{\dagger}\ M_m|\psi\rangle \tag{4.97}$$

In this equation, we substitute $\left[M_m^\dagger|x\right]$ and $[x|M_m]$ and conclude from

$$M_m^\dagger M_m = M_m M_m = M_m^2 = M_m \tag{4.98}$$

the equation for the probability $p\,(0)$ as follows.

$$\begin{aligned}
p\,(0) &= \langle\psi|M_0^\dagger\, M_0|\psi\rangle \\
&= \langle\psi|M_0|\psi\rangle \\
&= (\alpha^*\beta^*)\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} \\
&= (\alpha^*\beta^*)\begin{pmatrix}1\cdot\alpha+0\cdot\beta\\ 0\cdot\alpha+0\cdot\beta\end{pmatrix} \\
&= (\alpha^*\beta^*)\begin{pmatrix}\alpha\\ 0\end{pmatrix} \\
&= \alpha^*\cdot\alpha+\beta^*\cdot 0 \\
&= |\alpha|^2
\end{aligned} \tag{4.99}$$

We do the same procedure for $p\,(1)$ and calculate the probability to get the result $m = 1$.

$$\begin{aligned}
p\,(1) &= \langle\psi|M_1^\dagger\, M_1|\psi\rangle \\
&= \langle\psi|M_1|\psi\rangle \\
&= (\alpha^*\beta^*)\begin{pmatrix}0 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} \\
&= (\alpha^*\beta^*)\begin{pmatrix}0\cdot\alpha+0\cdot\beta\\ 0\cdot\alpha+1\cdot\beta\end{pmatrix} \\
&= (\alpha^*\beta^*)\begin{pmatrix}0\\ \beta\end{pmatrix} \\
&= \alpha^*\cdot 0+\beta^*\cdot\beta \\
&= |\beta|^2
\end{aligned} \tag{4.100}$$

$$p\,(m) = \begin{cases}|\alpha|^2\,, & \text{iff } I(m) = 0\\ |\beta|^2\,, & \text{iff } I(m) = 1\end{cases} \tag{4.101}$$

The calculations above show in a compact manner, that the result of a measurement assigns to 0 with probability $|\alpha|^2$, whereas it assigns to 1 with probability $|\alpha|^2$. As mentioned above, the system state changes after a measurement to the predecessing state ψ', respectively. Hence the system state after a measurement M_0 is shown below.

$$\begin{aligned}
|\psi'\rangle &= \frac{M_0\,|\psi\rangle}{\sqrt{\langle\psi|M_0^\dagger\,M_0|\psi\rangle}} \\
&= \frac{\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix}}{\sqrt{|\alpha|^2}} \\
&= \frac{\begin{pmatrix}\alpha\\ 0\end{pmatrix}}{|\alpha|} \\
&= \frac{\alpha\,|0\rangle}{|\alpha|} \\
&= \frac{\alpha}{|\alpha|}\,|0\rangle
\end{aligned} \tag{4.102}$$

To compute the system state after the measurement M_1, we compute as shown above with $m = 1$.

$$\begin{aligned}
|\psi'\rangle &= \frac{M_1\,|\psi\rangle}{\sqrt{\langle\psi|M_1^\dagger\,M_1|\psi\rangle}} \\
&= \frac{\begin{pmatrix}0 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix}}{\sqrt{|\alpha|^2}} \\
&= \frac{\begin{pmatrix}0\\ \beta\end{pmatrix}}{|\beta|} \\
&= \frac{\beta\,|1\rangle}{|\beta|} \\
&= \frac{\beta}{|\beta|}\,|1\rangle
\end{aligned} \tag{4.103}$$

After applying the measurement operators M_m on an arbitrary superposition of $|\psi\rangle$, we obtained the predecessing states after the transformation, $|\psi'\rangle$ as follows.

$$|\psi\rangle \to |\psi'\rangle = \begin{cases} \dfrac{\alpha}{|\alpha|}\,|0\rangle, & \text{iff } I(m) = 0 \\ \dfrac{\beta}{|\beta|}\,|1\rangle, & \text{iff } I(m) = 1 \end{cases} \tag{4.104}$$

Now let $|+\rangle$ and $|-\rangle$ be two states in a superposition. We know that the amplitudes $\alpha = \beta = \frac{1}{\sqrt{2}}$ are equal. Hence we get the result 0 with probability $p(0) = \frac{1}{2}$ and we get 1 with probability

$p\,(1) = \frac{1}{2}$.

$$p\,(0) = p\,(1) = |\alpha|^2 = |\beta|^2 = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2} \tag{4.105}$$

From this fact, we obtain that a measurement will result in an arbitrary value with probability $p\,(m)$. Hence for this two states, the computational basis $|+\rangle, |-\rangle$ are chosen.

$$p\,(m) = \begin{cases} \frac{1}{2}, & \text{iff } I(m) = 0 \\ \frac{1}{2}, & \text{iff } I(m) = 1 \end{cases} \tag{4.106}$$

After the measurement, the system state changes. Hence $|\psi'\rangle$ follows s.t.

$$\begin{aligned} |\psi\rangle \to |\psi'\rangle &= \frac{\alpha}{|\alpha|}\,|0\rangle = \frac{\frac{1}{\sqrt{2}}}{\left|\frac{1}{\sqrt{2}}\right|}\,|0\rangle = |0\rangle \\ |\psi\rangle \to |\psi'\rangle &= \frac{\beta}{|\beta|}\,|1\rangle = \frac{\frac{1}{\sqrt{2}}}{\left|\frac{1}{\sqrt{2}}\right|}\,|1\rangle = |1\rangle \end{aligned} \tag{4.107}$$

We see that for the two states holds.

$$\langle +|+\rangle = \langle -|-\rangle = 1 \quad \langle +|-\rangle = 0 \tag{4.108}$$

Hence the two states are orthonormal. From that fact we conclude, that the classic bit value b are mapped as follows

$$I\,(b) = \begin{cases} \text{“0”}, & \text{iff } |\psi\rangle = |+\rangle \\ \text{“1”}, & \text{iff } |\psi\rangle = |-\rangle \end{cases} \tag{4.109}$$

Furthermore, we get from M_m the measurement operators for the states $|+\rangle$ and $|-\rangle$.

$$M_0 = |+\rangle\langle +| = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \quad M_1 = |-\rangle\langle -| = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \tag{4.110}$$

By applying the measurement operators M_m and the states $|+\rangle$ and $|-\rangle$, we obtain the probability for the results for the probability of $m = 0$

$$p\,(0) = \langle +|M_0^\dagger\, M_0|+\rangle = \langle +|+\rangle\langle +|+\rangle = 1 \cdot 1 = 1, \tag{4.111}$$

and the probability of $m = 1$

$$p\,(1) = \langle +|M_1^\dagger\, M_1|+\rangle = \langle +|-\rangle\langle -|+\rangle = 0 \cdot 0 = 0. \tag{4.112}$$

In the next step, we calculate the state transformation from $|\psi\rangle$ to the state after the measurement $|\psi^{'}\rangle$. We obtain

$$|\psi\rangle \rightarrow |\psi^{'}\rangle = \frac{M_0\,|+\rangle}{\sqrt{\langle+|M_0^{\dagger}M_0|+\rangle}} = \frac{(|+\rangle\langle+|)\,|+\rangle}{\sqrt{\langle+|+\rangle\,\langle+|+\rangle}} = \frac{1}{1}\,|+\rangle = |+\rangle \tag{4.113}$$

Because the probability $p\,(0) = 1$, the state after the measurement assigns to $|+\rangle$. In the complementary case, the state for the result 1 does not assign to $|-\rangle$. Hence it is not defined.

Now we are applying the measurements of the states $|0\rangle$ and $|1\rangle$ in the computational basis $\{|+\rangle, |-\rangle\}$. The results for the probability $p\,(m)$ are shown below.

$$\begin{aligned}
p\,(0) &= \langle 0|M_0^{\dagger}M_0|0\rangle = \langle 0|+\rangle\,\langle +|0\rangle = \frac{1}{\sqrt{2}}\cdot\frac{1}{\sqrt{2}} = \frac{1}{2}\\
p\,(0) &= \langle 1|M_0^{\dagger}M_0|1\rangle = \langle 1|+\rangle\,\langle +|1\rangle = \frac{1}{\sqrt{2}}\cdot\frac{1}{\sqrt{2}} = \frac{1}{2}\\
p\,(1) &= \langle 0|M_1^{\dagger}M_1|0\rangle = \langle 0|-\rangle\,\langle -|0\rangle = \frac{1}{\sqrt{2}}\cdot\frac{1}{\sqrt{2}} = \frac{1}{2}\\
p\,(1) &= \langle 1|M_1^{\dagger}M_1|1\rangle = \langle 1|-\rangle\,\langle -|1\rangle = -\frac{1}{\sqrt{2}}\cdot -\frac{1}{\sqrt{2}} = \frac{1}{2}
\end{aligned} \tag{4.114}$$

Now we calculate the transformation from $|\psi\rangle$ to the successor state $|\psi^{'}\rangle$ as follows.

$$\begin{aligned}
|\psi\rangle \rightarrow |\psi^{'}\rangle &= \frac{M_0\,|0\rangle}{\sqrt{\langle 0|M_0^{\dagger}M_0|0\rangle}} = \frac{|+\rangle\,\langle+|0\rangle}{\sqrt{\langle 0|+\rangle\,\langle +|0\rangle}} = \frac{\frac{1}{\sqrt{2}}}{\frac{1}{\sqrt{2}}}|-\rangle = |+\rangle\\
|\psi\rangle \rightarrow |\psi^{'}\rangle &= \frac{M_0\,|1\rangle}{\sqrt{\langle 1|M_0^{\dagger}M_0|1\rangle}} = \frac{|+\rangle\,\langle+|1\rangle}{\sqrt{\langle 1|+\rangle\,\langle +|1\rangle}} = \frac{\frac{1}{\sqrt{2}}}{\frac{1}{\sqrt{2}}}|+\rangle = |+\rangle\\
|\psi\rangle \rightarrow |\psi^{'}\rangle &= \frac{M_1\,|0\rangle}{\sqrt{\langle 0|M_1^{\dagger}M_1|0\rangle}} = \frac{|-\rangle\,\langle-|0\rangle}{\sqrt{\langle 0|-\rangle\,\langle -|0\rangle}} = \frac{\frac{1}{\sqrt{2}}}{\frac{1}{\sqrt{2}}}|-\rangle = |-\rangle\\
|\psi\rangle \rightarrow |\psi^{'}\rangle &= \frac{M_1\,|1\rangle}{\sqrt{\langle 1|M_1^{\dagger}M_1|1\rangle}} = \frac{|-\rangle\,\langle-|1\rangle}{\sqrt{\langle 1|-\rangle\,\langle -|1\rangle}} = \frac{-\frac{1}{\sqrt{2}}}{\frac{1}{\sqrt{2}}}|-\rangle = (-1)\cdot|-\rangle = |-\rangle
\end{aligned} \tag{4.115}$$

From that we conclude, that a measurement in the wrong basis results in a randomly result with probability $p\,(m) = \frac{1}{2}$.

4.3 Basic Model

Quantum key distribution (QKD) was first presented in 1984, by Bennett and Brassard [3]. It is a method based on the quantum laws of physics. The basic model of QKD is given in figure 4.5. It is not a technique, to encrypt and decrypt a plaintext. However, QKD solves the problem of key distribution between Alice (the sender) and Bob (the receiver). Based on the quantum laws of

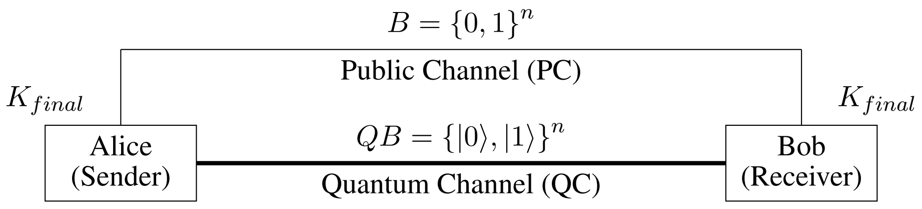


Figure 4.5: Principal Model of Quantum Key Distribution

physics, it enables provable [44] unconditional security and processes the generation of a random shared secret K_{final} between two communication parties. Unconditional security means that the level of security does not rely on unproven computational complexity assumptions. Hence it does not matter how much computation power or time is available to an arbitrary eavesdropper. Furthermore, an unconditional secure ciphertext can not be broken, because the cipher does not provide any information to determine the plaintext uniquely. QKD complements classic algorithms for authentication and error detection or error correction. Single photon pulses i.e. qubits $(QB = \{|0\rangle, |1\rangle\}^n)$ are transmitted over an authenticated but noisy quantum channel, whereas the classic bits are transmitted over public (probably unsecure) channels $(B = \{0, 1\}^n)$. That means the classical transmission over the public channel can be interpreted as a ciphertext, a plaintext or parity bits for error handling for instance. The main idea of quantum key distribution is, to provide eavesdropping detection.

4.4 Quantum Channel vs. Public Channel

A communication channel is an arbitrary physical implementation, which allows two communication parties, to exchange any information. In quantum information theory, a quantum channel is implemented as a single fibre access link. Hence it is a peer to peer connection between two communication parties. Furthermore, a quantum channel is a communication channel, which enables the transmission of quantum information, to provide a private communication of quantum information. The smallest unit in quantum information is similar to the logical bit, in classic information theory, the qubit. Qubits are used to transport the information on the quantum channel from Alice to Bob. Classic information is a plaintext or a ciphertext $X, Y \in \{0, 1\}^n$, described by an $n-$tuple, for instance.

4.5 BB84

In 1984, Charles Bennett and Gilles Brassard (BB84) proposed the first quantum key distribution protocol. Because of the properties of quantum states, that we cannot duplicate an arbitrary, unknown quantum state (no cloning theorem), BB84 is provable secure. The BB84 protocol is based on three primary stages, the key exchange, key sifting and key distillation stage. Furthermore we can granulite the key distillation stage into three secondary phases (i.e. error estimation and error correction, privacy amplification and authentication).

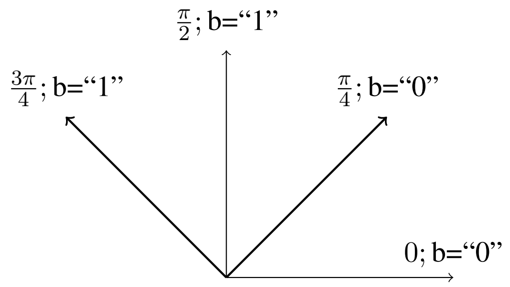
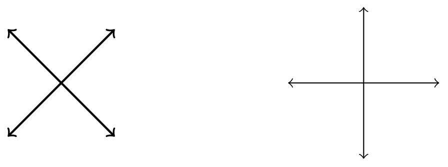


Figure 4.6: Bit Encoding of the Photon Polarizations Quantum Key Distribution



(a) Diagonal Basis (b) Rectilinear Basis

Figure 4.7: Bit Encoding of the Measurements in Quantum Key Distribution

1 1 1 0 0 0 1 0 0 1 0 1 0 1 1 0

Figure 4.8: BB84 Random Bit Sequence chosen by Alice

The first primary stage establishes the transmission of a random encoded single photon stream of size n by Alice (the sender) to Bob (the receiver). Every photon can be encoded in four polarization angles $\left(\text{i.e. } 0,\ \frac{\pi}{4},\ \frac{\pi}{2},\ \frac{3\pi}{4}\right)$ as shown in figure 4.6. Furthermore it can be represented by the conjugate bases rectilinear and diagonal (i.e. $\oplus = \{0, \frac{\pi}{2}\}$ and $\otimes = \{\frac{\pi}{4}, \frac{3\pi}{4}\}$) as shown in figure 4.7. That means a single photon can be prepared by a measurement in the rectilinear or diagonal basis. The bases are conjugate because a measurement in the complementary basis results in random information. However, the pairing of orthogonal couples will be represented as classic bit values "0" and "1" e.g. 0 and $\frac{\pi}{4}$ corresponding to "0" and the remaining angles $\frac{\pi}{2}$ and $\frac{3\pi}{4}$ corresponding to "1".

Alice invokes the function `(Q)URAND` and generates a (quantum) random bit stream (B), which is truely random. The random bit stream is shown in Fig. 4.8. She chooses a random polarization sequence (PS) and combines the bit stream with the random measurement preparations mentioned in Fig. 4.6 and 4.7. Figure 4.9 illustrates the measurement preparations which are kept secret, respectively. It shows the locations at which the polarization filter of Alice does not match the polarization filter of Bob.

The results of the measurements are single polarized photons which are transmitted on the quantum channel from Alice to Bob and which are denoted as (QB) and specify a stream of qubits. The measurements in the corresponding basis are shown in Fig. 4.10 and Fig. 4.11. Bob also prepares its random polarization sequence to measure the incoming photons on the quantum channel. If Bob measures in the correct basis i.e. the same as Alice chose, he obtains a valid bit

$$\oplus \; \otimes \; \otimes \; \oplus \quad \otimes \; \otimes \; \oplus \; \oplus \quad \otimes \; \otimes \; \oplus \; \oplus \quad \otimes \; \oplus \; \otimes \; \oplus$$
$$\oplus \; \otimes \; \oplus \; \otimes \quad \oplus \; \otimes \; \otimes \; \otimes \quad \otimes \; \oplus \; \oplus \; \otimes \quad \oplus \; \oplus \; \otimes \; \oplus$$

Figure 4.9: Random Polarization of Alice and Bob in BB84

Basis $\oplus$	Information							
b	1	0	1	0	1	1	0	1
φ	$\frac{\pi}{2}$	0	$\frac{\pi}{2}$	0	$\frac{\pi}{2}$	$\frac{\pi}{2}$	0	$\frac{\pi}{2}$

Figure 4.10: Measurement of 1 B to the Basis $\oplus$ in BB84

Basis $\otimes$	Information							
b	1	0	1	0	1	1	0	1
φ	$\frac{3\pi}{4}$	$\frac{\pi}{4}$	$\frac{3\pi}{4}$	$\frac{\pi}{4}$	$\frac{3\pi}{4}$	$\frac{3\pi}{4}$	$\frac{\pi}{4}$	$\frac{3\pi}{4}$

Figure 4.11: Measurement of 1 B to the Basis $\otimes$ in BB84

value, otherwise the bit value will be random with probability $\frac{1}{2}$. After that Bob obtains its raw key K^B_{RAW} and its length is at most n_0. The length of the streams can decrease iteratively from n_0 to n_5 because of noise on the quantum channel, otherwise the length of the raw keys is equal.

Figure 4.12 illustrates the distinction on the quantum key exchange to which stages are transmitted on the quantum channel and which are transmitted on the public channel. It shows that the quantum channel provides only quantum encoding and the preprocessing and postprocessing methods are based on classical transmissions over the public channel. Figure 4.13 shows the

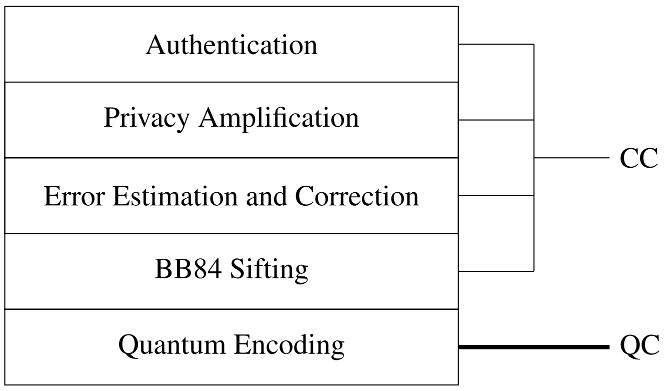


Figure 4.12: Protocol Flow of the Quantum Key Exchange

sequence diagram of primary protocol stages of the BB84 quantum protocol to derive a raw key between Alice and Bob. It shows that the quantum transmission is a unidirectional transmission where Alice denotes the sender and Bob denotes the receiver. The qubits are to be transmitted on the noisy quantum channel and the classical bits are transmitted on the probably unsecure

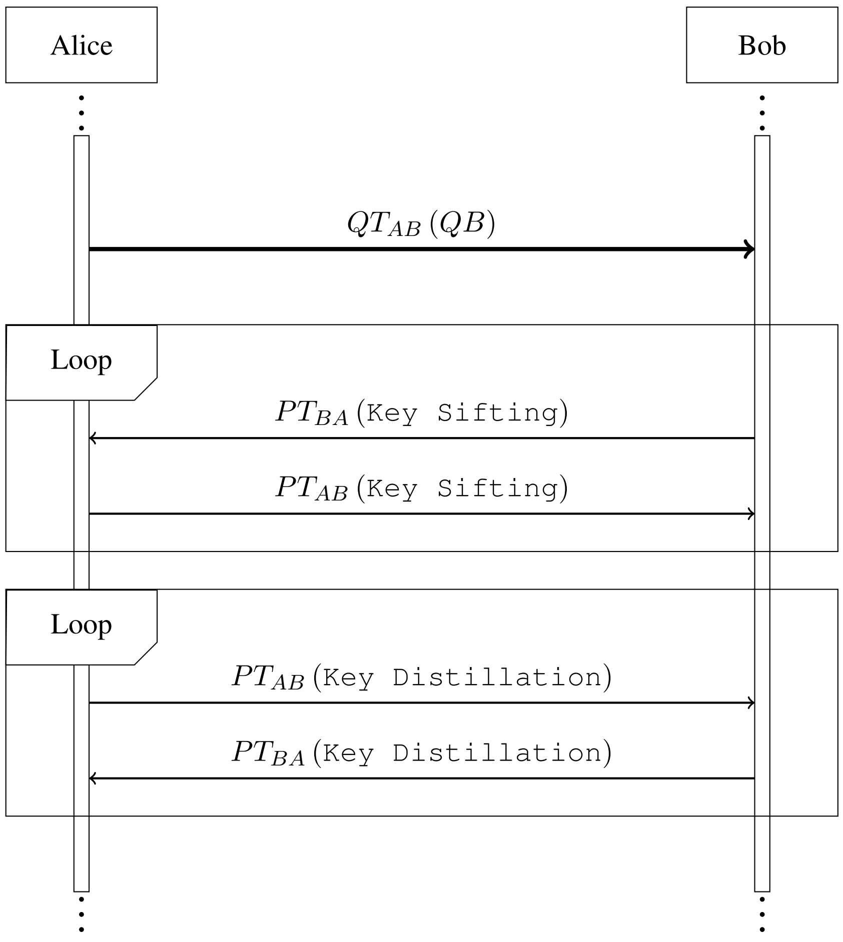


Figure 4.13: Sequence Diagram of the Primary Protocol Stages of BB84

public channel. If we want to derive a raw key by sending qubits over the quantum channel from Alice to Bob, we have to preprocess and postprocess before the transmission of the qubits and after it. The preprocessing and postprocessing techniques to derive a raw key are shown in Fig. 4.14.

The calculation rounds from the raw key generation and the calculation of the sifted key are summarized in Fig. 4.15. It illustrates that the key generation is based on a random bit stream, Alice calculates and a polarization measurement to generate the transmission sequence for the transmission on the quantum channel i.e. Alice generates a random stream of qubit to send them to Bob. Bob measures the qubits on the quantum channel and gains polarizations and calculates the bit values for the raw key.

After the transmission over the quantum channel, the result will be an initial raw key material on both sides with at most n bits during noise on the quantum channel. In summarization, this may also be referred to as the key exchange or raw key exchange (RKE) between two communication parties.

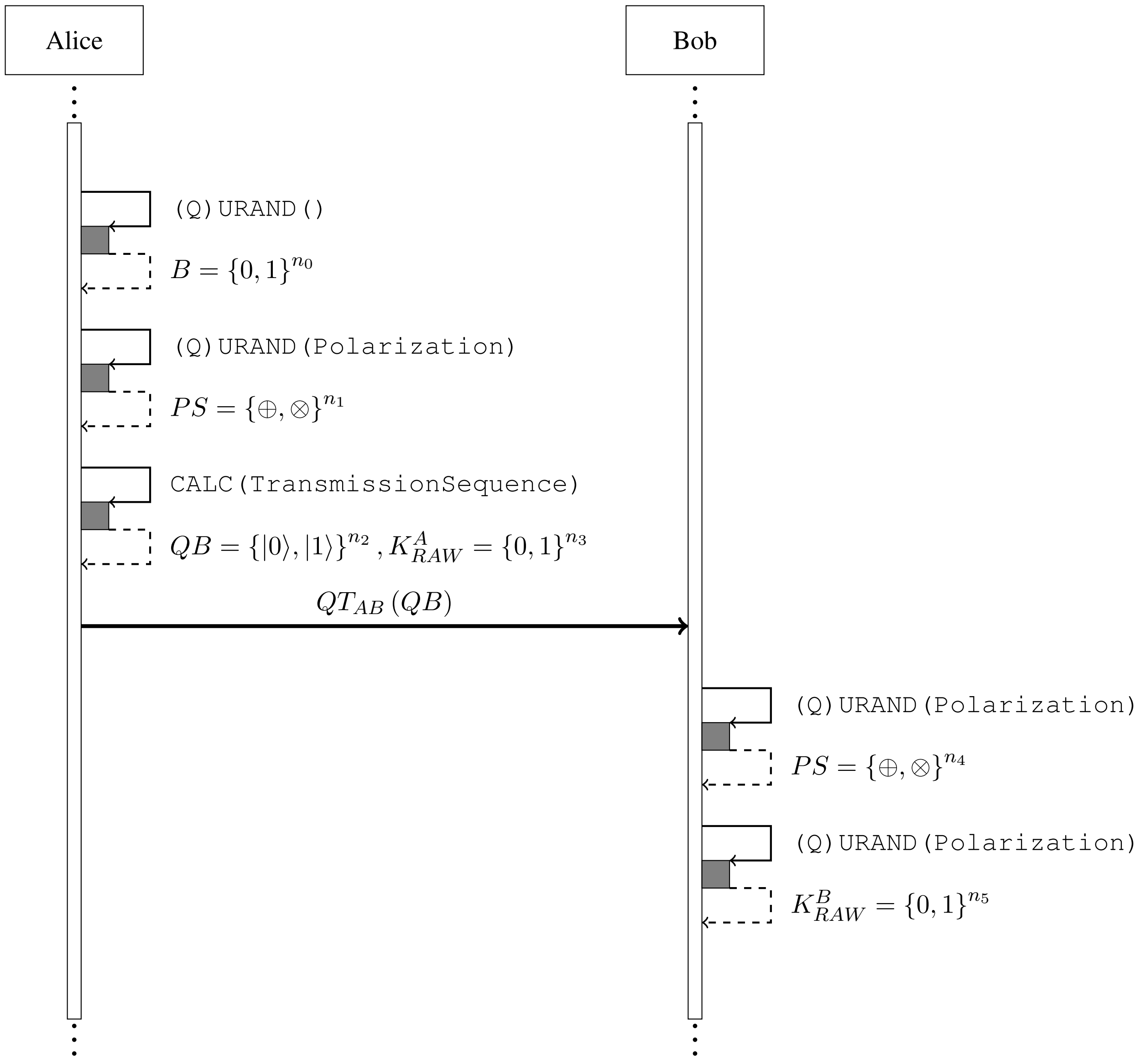

Alice
Bob
(Q)URAND()
$B = \{0,1\}^{n_0}$
(Q)URAND(Polarization)
$PS = \{\oplus, \otimes\}^{n_1}$
CALC(TransmissionSequence)
$QB = \{|0\rangle, |1\rangle\}^{n_2}, K^A_{RAW} = \{0,1\}^{n_3}$
$QT_{AB}(QB)$
(Q)URAND(Polarization)
$PS = \{\oplus, \otimes\}^{n_4}$
(Q)URAND(Polarization)
$K^B_{RAW} = \{0,1\}^{n_5}$

Figure 4.14: Sequence Diagram of the Quantum Transmission to obtain the Raw Keys (K^A_{RAW}, K^B_{RAW}) in BB84 - $n_i \leq n_{i+1}$

							Bob				
				Alice							
					Transmission Sequence						
0	1	$\oplus$	$\vert$	$\frac{\pi}{2}$	$\|1\rangle$	$\downarrow$	$\oplus$	$\frac{\pi}{2}$	$\downarrow$	1	1
1	1	$\otimes$	$\diagdown$	$\frac{3\pi}{4}$	$\|-\rangle$	$\leftarrow$	$\otimes$	$\frac{3\pi}{4}$	$\leftarrow$	1	1
2	1	$\otimes$	$\diagdown$	$\frac{3\pi}{4}$	$\|-\rangle$	$\leftarrow$	$\oplus$	0	$\uparrow$	0	$*$
3	0	$\oplus$	—	0	$\|0\rangle$	$\uparrow$	$\otimes$	$\frac{3\pi}{4}$	$\leftarrow$	1	$*$
4	0	$\otimes$	$\diagup$	$\frac{\pi}{4}$	$\|+\rangle$	$\rightarrow$	$\oplus$	$\frac{\pi}{2}$	$\downarrow$	1	$*$
5	0	$\otimes$	$\diagup$	$\frac{\pi}{4}$	$\|+\rangle$	$\rightarrow$	$\otimes$	$\frac{\pi}{4}$	$\rightarrow$	0	0
6	1	$\oplus$	$\vert$	$\frac{\pi}{2}$	$\|1\rangle$	$\downarrow$	$\otimes$	$\frac{\pi}{4}$	$\rightarrow$	0	$*$
7	0	$\oplus$	—	0	$\|0\rangle$	$\uparrow$	$\otimes$	$\frac{\pi}{4}$	$\rightarrow$	0	$*$
8	0	$\otimes$	$\diagup$	$\frac{\pi}{4}$	$\|+\rangle$	$\rightarrow$	$\otimes$	$\frac{\pi}{4}$	$\rightarrow$	0	0
9	1	$\otimes$	$\diagdown$	$\frac{3\pi}{4}$	$\|-\rangle$	$\leftarrow$	$\oplus$	$\frac{\pi}{2}$	$\downarrow$	1	$*$
10	0	$\oplus$	—	0	$\|0\rangle$	$\uparrow$	$\oplus$	0	$\uparrow$	0	0
11	1	$\oplus$	$\vert$	$\frac{\pi}{2}$	$\|1\rangle$	$\downarrow$	$\otimes$	$\frac{3\pi}{4}$	$\leftarrow$	1	$*$
12	0	$\otimes$	$\diagup$	$\frac{\pi}{4}$	$\|+\rangle$	$\rightarrow$	$\oplus$	0	$\uparrow$	0	$*$
13	1	$\oplus$	$\vert$	$\frac{\pi}{2}$	$\|1\rangle$	$\downarrow$	$\oplus$	$\frac{\pi}{2}$	$\downarrow$	1	1
14	1	$\otimes$	$\diagdown$	$\frac{3\pi}{4}$	$\|-\rangle$	$\leftarrow$	$\otimes$	$\frac{3\pi}{4}$	$\leftarrow$	1	1
15	0	$\oplus$	—	0	$\|0\rangle$	$\uparrow$	$\oplus$	0	$\uparrow$	0	0

Figure 4.15: Example Key Exchange of Alice and Bob in BB84

The second stage is the BB84 sifting phase. Sifting means that Alice and Bob reveal information about the sent and detected stream over an unsecure channel e.g. the internet. However in practice, the unsecure channel is a result of a switch to a pseudo-classic channel on the physical implementation of the quantum channel by wave division multiplexing. Because a single photon can be measured only once, we can apply one basis on it. Yet the photon can be measured in two different types. On the one hand to the correct base, on the other hand, to the wrong base. The measurement with the correct base will result in a unique measurement value. If the photon is measured in the wrong basis, the result will be a random value with probability $p\,(m) = \frac{1}{2}$. Alice maintains only the polarization values, received by Bob in the correct basis and sends this revised list of polarization values back to Bob over the classic channel. Alice and Bob get a revised alignment of polarization values, where the measured values are equal. This alignment is also called the sifted key and is of equal size for both communication parties i.e. n bits. The result of the computation of the sifted key is shown in Fig. 4.16 where $*$ denotes the purged bits. That means that Alice and Bob chose a divergent polarization during the measurement and the corresponding bits are eliminated. However, this alignment is not necessarily complete and may contain errors of unknown quantity. The sifted key rate (SKR) is shown in the following

$$1 \quad 1 \quad * \quad * \qquad * \quad 0 \quad * \quad * \qquad 0 \quad * \quad 0 \quad * \qquad * \quad 1 \quad 1 \quad 0$$

Figure 4.16: Example Key Exchange to obtain the Sifted BB84 Key Material in BB84

equation.

$$m = n \cdot \mu \cdot \alpha_f \cdot \alpha_p \cdot \alpha_e \cdot \eta_{det} \cdot k_{dead} \tag{4.116}$$

It is taken from [50]. Where n denotes the sifted key length. The number of photons at the transmitter by Alice is denoted by μ. Because Alice and Bob choose their bases at random, the length of the raw key is less or equal than the sifted key length. A quantum channel is a noisy channel, hence it has fibre losses per distance α_f. The additional loss of the system is denoted by α_e. Corresponding to the protocol rounds of the implemented protocol, the losses of it effect on the SKR which are denoted by α_p and are in the interval $[0;1]$. If BB84 is implemented the losses will be $\frac{1}{2}$. By the use of quantum measurements, the detector efficiency η_{det} and the dead time factor k_{dead} reduce on the SKR, respectively.

The property of the decreasing length of streams and erroneous bits show the need for key distillation techniques to handle the errors in the raw keys and the divergent length, respectively. Error handling is a method to detect errors in the raw key and eliminate them iteratively to obtain a valid key from the sifted key. Figure 4.17 shows a sequence diagram for the key distillation phase and its three stages. Privacy amplification and authentication are not in the scope of this work. More information on privacy amplification and authentication can be found in [2, 4, 16]. The stages from the corresponding error handling are shown in Fig. 4.18. Alice and Bob store a temporary database about their polarization information to set up an iterative comparison of the sets of polarization filters for the subsequent sifting phase.

The third stage is the key distillation phase. It will be divided into three secondary stages (error correction, privacy amplification and authentication). The erroneous sifted key will be post-processed to obtain the reconciled key, hence this stage is called error correction phase or reconciliation. During the reconciliation stage, a set of parity bits and error correction codes are exchanged between Alice and Bob. The task is to avoid the transmission of explicit key values over the unsecure channel and to eliminate the errors from the sifted key. However, the reconciled key is smaller or equal in the length of bits, than the sifted key from the stage before. After obtaining the reconciled key from the sifted key it is necessary to process the next stage to obtain a private key bit stream. Nevertheless an eavesdropper can gain information about the reconciled key via the intercept and resend strategy or the use of a beamsplitter. If an eavesdropper uses the intercept and resend strategy on a sufficient small number of bits, the induced errors will be lost among the errors which are introduced by the noisy quantum channel, detector noise and other physical problems caused by the implementation. The problem of detecting an eavesdropper is identical to the problem of error correction on the quantum channel. Hence it is not possible to distinguish whether an error is based on corruptive behavior or the result of the physical implementation. We summarize it as noise as a result of eavesdropping activity on the quantum channel. Since we know that the filter polarization on Alice and Bob's side was random for each bit, we know that the initial bitstream was random i.e. the sifted key was random. Hence the reconciled key is random and after applying the privacy amplification

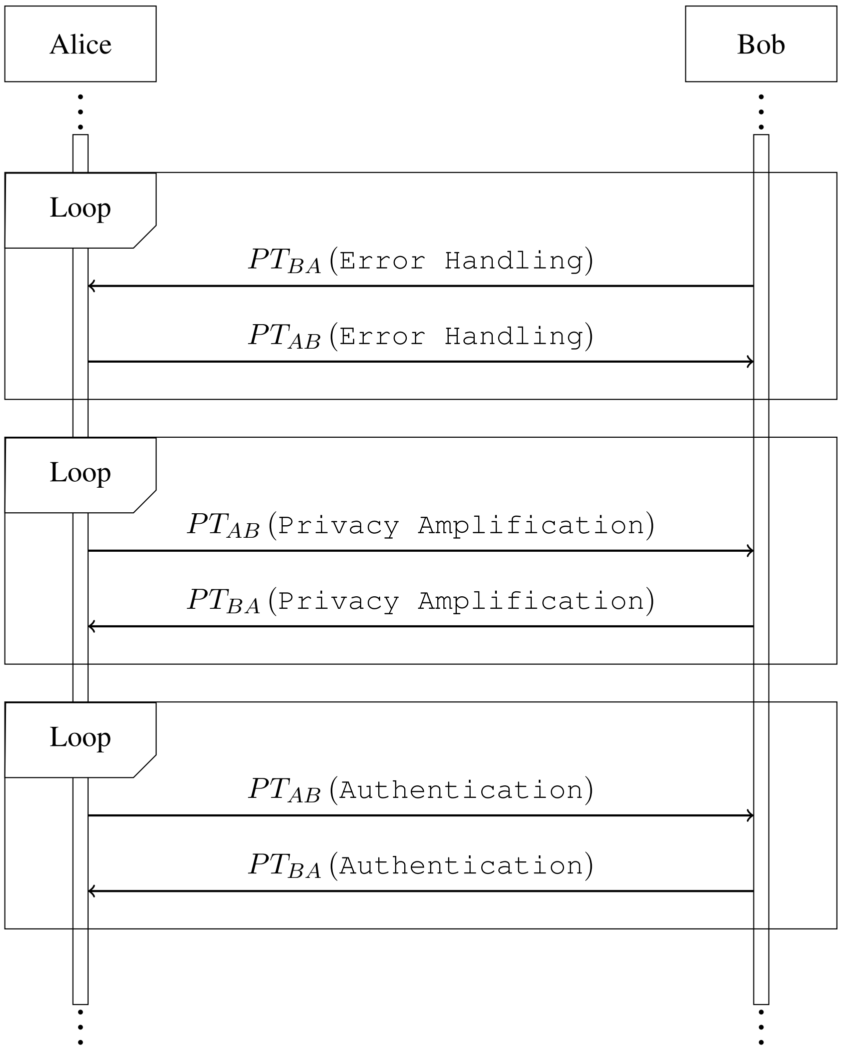

Alice
Bob
Loop
PT_{BA} (Error Handling)
PT_{AB} (Error Handling)
Loop
PT_{AB} (Privacy Amplification)
PT_{BA} (Privacy Amplification)
Loop
PT_{AB} (Authentication)
PT_{BA} (Authentication)

Figure 4.17: Sequence Diagram of the Key Distillation in BB84

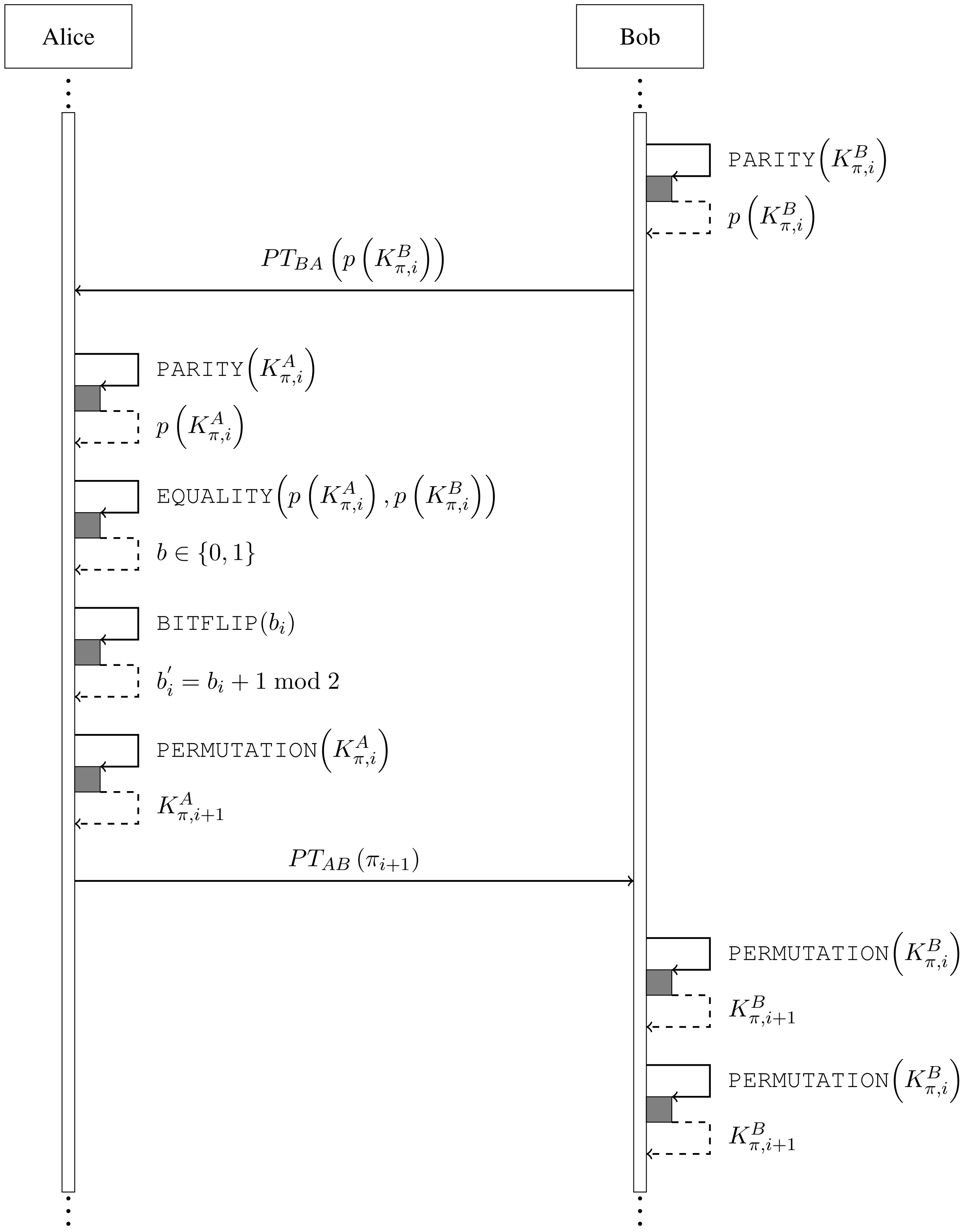

Alice
Bob
PARITY$\left(K^B_{\pi,i}\right)$
$p\left(K^B_{\pi,i}\right)$
$PT_{BA}\left(p\left(K^B_{\pi,i}\right)\right)$
PARITY$\left(K^A_{\pi,i}\right)$
$p\left(K^A_{\pi,i}\right)$
EQUALITY$\left(p\left(K^A_{\pi,i}\right), p\left(K^B_{\pi,i}\right)\right)$
$b \in \{0,1\}$
BITFLIP(b_i)
$b_i' = b_i + 1 \bmod 2$
PERMUTATION$\left(K^A_{\pi,i}\right)$
$K^A_{\pi,i+1}$
$PT_{AB}(\pi_{i+1})$
PERMUTATION$\left(K^B_{\pi,i}\right)$
$K^B_{\pi,i+1}$
PERMUTATION$\left(K^B_{\pi,i}\right)$
$K^B_{\pi,i+1}$

Figure 4.18: Sequence Diagram of the Transmission of parity bits over the Public Channel in the BB84 Error Handling

hash algorithm, the privacy amplified key is random, too.

Let us assume that Alice chooses one single bit with value 1. Remember the bit encodings from Fig. 4.7 and Fig. 4.6 and assume that Alice has chosen the horizontal/vertical $\oplus$ polarization filter, which results in the angle $\frac{\pi}{2}$ and transmits it on the quantum channel. The eavesdropper (Eve) uses the intercept and resend strategy whereas Bob choose the correct basis (i.e. $\oplus$) for his measurement. Eve has no knowledge about the chosen polarization by Alice and Bob and uses its own random pattern of polarization filters (i.e. $\oplus$ or $\otimes$). After that Eve re-sends the measurement results, where Bob receives either $\frac{\pi}{4}$ or $\frac{3\pi}{4}$ on the quantum channel with probability $\frac{1}{2}$. An overview of the output of random bit values if an eavesdropper listens to the quantum channel and resends qubits by its random pattern is given in Fig. 4.19. In the upper polarization preparation Eve choose the wrong basis, otherwise Eve choose the correct basis. It illustrates that eavesdroppers can be detected referencing to the randomly chosen bases taken by Alice and Bob by comparing them iteratively. A sufficiently successful strategy if Eve has been eavesdropping

Alice			Eve	Bob		
				$\frac{\pi}{4}$	$\oplus$	0, $\frac{\pi}{2}$
1	$\oplus$	$\frac{\pi}{2}$	$\otimes$			
				$\frac{3\pi}{4}$	$\oplus$	0, $\frac{\pi}{2}$
				0	$\otimes$	$\frac{\pi}{4}$, $\frac{3\pi}{4}$
0	$\otimes$	$\frac{\pi}{2}$	$\oplus$			
				$\frac{\pi}{2}$	$\otimes$	$\frac{\pi}{4}$, $\frac{3\pi}{4}$

Figure 4.19: Eavesdropping Intercept and Resend on the Quantum Channel

on the quantum channel, is the exchange of the alignment of the polarization directions on the public channel between Alice and Bob. That means after Alice has sent a random stream of qubits QB on the quantum channel by polarizing them with a random polarization sequence, to share the same key with Bob. The sifting phase on the public channel is done via the transmission of the polarization configuration. Now Alice knows the bits where Bob has chosen the correct basis for the measurement. If Bob measures in the correct basis, the bit value is valid, otherwise it is invalid and will be eliminated. Losses on the quantum channel and node internal detector efficiency affect on the transmission of the qubits i.e. sometimes Bob is not able to measure a photon. The necessary information to complete the sifting phase is stored in quantum tables and base tables. If there are no clicks in the bases of Alice and Bob, the base will be set invalid. In the complementary case the base is set to diagonal or rectilinear. This corresponds to the detector click. Now the base tables are compared to find out misalignments. Bob sends its base table to

Alice who compares the values and sets all bases to valid if they have chosen the same basis. It is set invalid, otherwise. Due to the physical implementation it occurs that double detector clicks exist because of synchronization mismatches. If a double click occurs generate a random bit value and write it to the quantum table. This introduces the Quantum Bit Error Rate (QBER) for noisy quantum channels and is a valid indicator for an eavesdropping attack on the QC. Alice chooses a random bit sequence of size $\frac{m}{2}$ which she wants to reject, e.g. as shown in Fig. 4.20. This strategy results in the remaining shared secret between Alice and Bob after the elimination

$$0 \quad 1 \quad 1 \quad 0$$

Figure 4.20: Random Bit Rejection of the Example Key Exchange in BB84

of $\frac{m}{2}$ bits. After the rejection of the random bits Alice and Bob obtain the final key for the unconditional secure communication. Because this key is based on (quantum) randomness they use it only once as a one time pad to encrypt a ciphertext unconditionally secure at most in the length of the final key. That means in practice that the message to transmit encrypted is as long as the quantum key. The final quantum key is illustrated by Fig. 4.21. Figure 4.22 shows the

$$1 \quad 1 \quad 0 \quad 0$$

Figure 4.21: Final Shared Secret of the Example Key Exchange in BB84

protocol gain of the BB84 protocol if the probability to choose a basis differs from the low point $\frac{1}{2}$. The formula is taken from [32].

$$\begin{aligned} g_p &= \delta \cdot \delta + (1-\delta) \cdot (1-\delta) \\ &= 2\delta^2 - 2\delta + 1 \end{aligned} \tag{4.117}$$

That means we can vary the probability to choose a basis diagonal or rectilinear during the calculation rounds to derive the raw key. The problem will be that Alice and Bob have to agree on a fixed value to which probability the corresponding basis will be chosen during the key agreement stage. That means some information on the key is given to the eavesdropper. This information reduces the security of the chosen protocol.

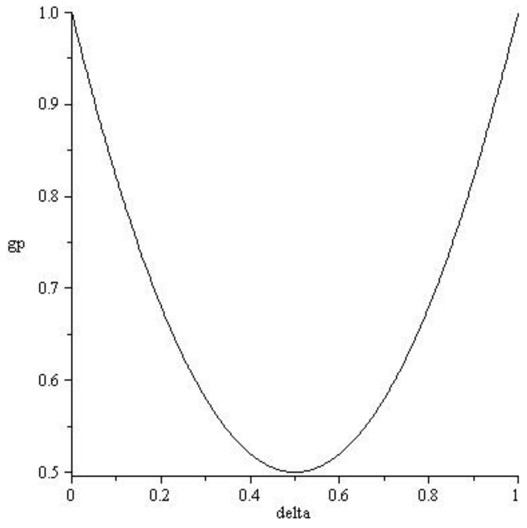


Figure 4.22: Maple Simulation of the Protocol Gain for variable Probability to choose a Polarizationfilter in BB84 - g_p

CHAPTER 5

Securing IPv6 by Quantum Key Distribution

The internet protocol IPv6 provides an end to end connectivity over long distances. Quantum key distribution is a method which provides unconditional secure secret key exchange between two distant communication nodes. However, QKD is limited in the distance and the key generation rate, respectively. Undconditionally secure quantum repeaters which propagate light quantas over large distances are not invented, yet. Hence we have to combine QKD with classical solutions to bridge larger distances. This chapter describes the classic point to point connection of QKD nodes and shows that a noisy quantum channel has losses which depend on the length of the optical fibre. It will show that quantum networks solve the problem of key distribution between two distant nodes even if there are unreliable nodes in the quantum network. That means quantum networks are based on multiple point to point QKD connections. This section will show that not every intermediate device needs to be reliable to enable unconditional security between Alice and Bob.

5.1 Security Perimeter

An approach for security perimeters are firewalls which seperate demilitarized zones (DMZ) e.g. from the internet or various zones of different security levels, as known as gateways. A set of policies manage the use of IT infrastructure in probably unsecure environments and avoid from internal risk potential and external threats. We define that the interprocess communication on both machines and the wired connections between the machines and the router and the QKD hardware are unconditionally secure. Furthermore we need an authenticated quantum channel to guarantee unconditionally secure quantum key exchange. That means an eavesdropper can listen on the quantum channel and it can receive and resend qubits.

5.2 QKD Networks

In the classical way, a provable secure quantum connection can only be established between two adjacent nodes as a point to point connection. Because the transmission rate is indirect proportional to the transmission distance, we need a method to put long distances for QKD applications and to ensure adequate transmission rates. Today we have no technique to repeat qubits securely on a quantum channel as similar to a WiFi repeater in the classical way. IPv6 is a modular communications protocol to enable the data processing over long distances with a point to point connection and the involvement of security technology to provide secrecy. Today we can not transmit qubits over distances greater than 100km efficiently. To solve the problem of transmission rate and distance we specify quantum networks.

QKD Point to Point Interconnection

A QKD point to point (p2p) Connection is similar to a classic p2p connection by contrast the additional existence of a lossy quantum channel (QC) between Alice and Bob. QC's are not used to send ciphertext or plaintext information, yet but QC's are used to transmit manipulated, random quantumbits (qubits) between Alice and Bob. Hence no initial shared secret is necessary to establish secure communication, per definition. A QC transmits quantum information i.e. qubits over a noisy but authenticated channel. Hence in QKD systems, a p2p connection is established by two channels virtually, the quantum channel and the public channel (PC). Furthermore wave division multiplexing (WDM) is used to switch between the public (classic) and quantum channel and multiplexes/demultiplexes the pulse sequence, virtually.

The physical implementation belongs to an optical channel, which denotes a physical link between Alice and Bob to transmit photons on the quantum channel i.e. the qubits are transmitted over the quantum channel. A laser from a quantum router sends photon pulses. However, optical modems polarize the photons, which encode the quantum information. The photon detector receives the quantum information and performs the measurement on Bob's side. Quantum protocols generate unconditional secure keys with pre- and postprocessing methods to deliver an arbitrary, final shared secret between Alice and Bob. Hence QKD denotes a Quantum Key Distribution System and delivers key material for various encryption techniques from one point to another. Figure 5.1 shows the main idea behind the quantum point to point concept, between Alice (A) and Bob (B). The 2 adjacent QKD nodes (Alice, Bob) are connected via a quantum channel and make use of a wave division multiplexing, from one endpoint to another.

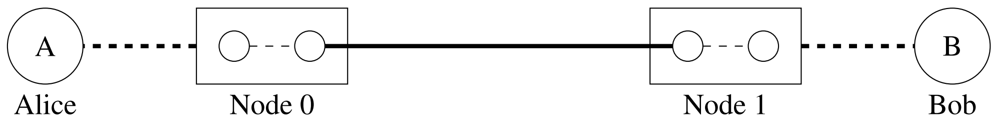


Figure 5.1: Quantum Point to Point Interconnection Model

Transmission Loss

A quantum channel is implemented as a noisy optical fibre. Hence there are transmission losses during a quantum transmission which are introduced by the physics of the quantum channel itself. However, this does not separate quantum connections from connections in the classical way but it shows that QKD is limited yet in the distance and speed. The transmission loss t_L on an optical fibre between Alice and Bob is shown in the following equation. The formula is taken from [32].

$$t_L = 10^{-\frac{\alpha \cdot d}{10}} \tag{5.1}$$

The variable α denotes the absorption on the quantum channel in dB/km and d denotes the distance in m. The transmission loss on the quantum channel is shown in figure 5.2. It shows

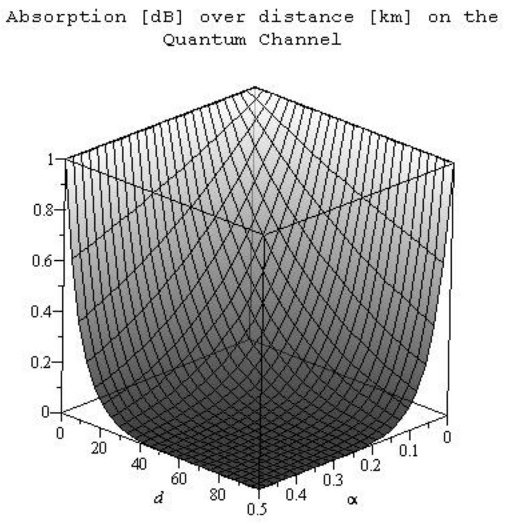


Figure 5.2: Maple Simulation of the Absorption on the Quantum Channel - α, d variable

the behavior of a quantum channel when α, d are variable. To show the behavior of a real optical fibre, we construct an instance of a quantum channel at different but fixed losses and variable length of the channel. Usual optical fibres in practical use have losses in the amount of $\alpha \in \{0.2, 0.3, 0.33, 0.5\}$ dB/km, respectively. These values are in the area of a wavelength of $\lambda \approx 1\,500\,\text{nm}$ and optical fibres with 0.2 dB/km are called low-attenuation optical fibres. The wavelength λ is a value of wave where the pattern of the wave repeats. That means it can be measured between two zero-crossings or the maximum amplitudes of a function of the same phase. A simulation of the instance is given in Fig. 5.3 below. It shows that the transmission is limited to the length of the channel and the losses by the physical implementation of the quantum channel. Because of these limits, QKD is not an ideal solution for applications over wide areas. However, if we want to use it for point to point communications over the internet and construct a quantum extension to the modular IPv6 communications protocol for applications between two distant nodes, we need additional concepts which rely on classical assumptions and do not

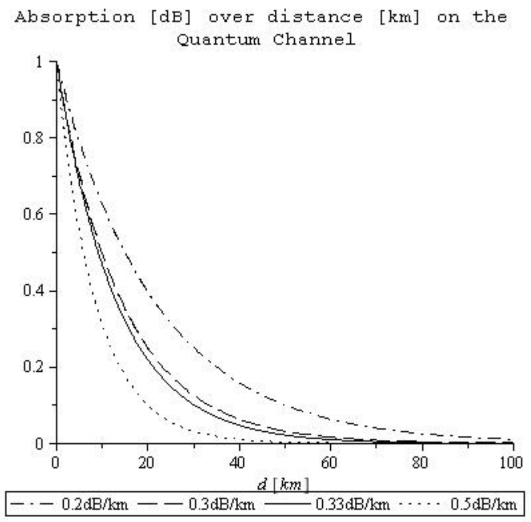


Figure 5.3: Maple Simulation of the Absorption on the Quantum Channel - d variable

belong to the physics of QKD, yet. Figure 5.4 shows the rates of generation of raw key material with the BB84 protocol, it is taken from [50]. The raw key generation rate is given by Eq. (5.2).

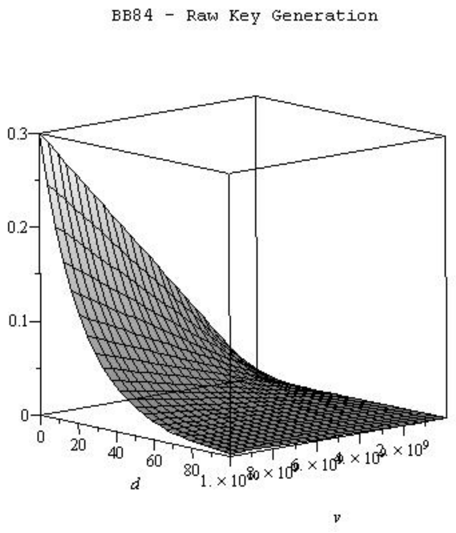


Figure 5.4: Maple Simulation of the Raw Key Generation K_{RAW} - ν, d variable

$$K = q \cdot \nu \cdot \mu \cdot t_L \cdot t_B \cdot \eta_B \cdot \eta_T \tag{5.2}$$

where q depends on the implemented protocol and for BB84 $q = \frac{1}{2}$. The repetition frequency ν is in the interval $[0.001; 10]$GHz and the distance between $[0; 100]$km are variable. The simulation shows the results for a fixed average number of photons per pulse e.g. $\mu = 0.1$ and a transmission loss α of 0.2 dB/km as shown in Eq. (5.1) and Fig. 5.3. We also fix the internal transmission factor of Bob $(t_B)$by 0.6 and set its detection efficiency (η_B) to 0.1. The detector dead time (ν_T) is fixed and set to 10 ns.

Quantum Network Interconnection

To establish a connection over wide areas i.e. in the case of a quantum connection greater than 100 km we specify the QKD network interconnection model to bridge larger distances and build a connection between two distant nodes. It is based on the laws of quantum physics and enables unconditional security over large distances, if a few requirements are satisfied. Hence a QKD Network Interconnection model is similar to a QKD p2p connection and can be interpreted as a quantum backbone network infrastructure (QBNI). A backbone is usually an optical fibre endpoint that can be connected to an external network i.e. the quantum network and to a node internal network.

Figure 5.5 shows a single path with 2 intermediate nodes (Node 1, Node 2) between Alice (Node 0) and Bob (Node 3) to increase the distance, respectively. To encrypt an arbitrary plain-

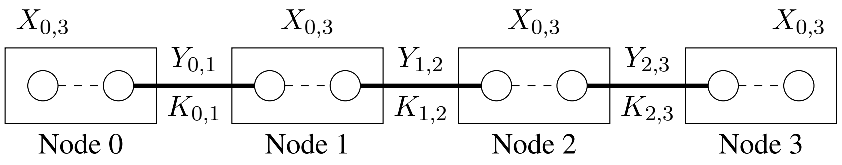


Figure 5.5: Transmission of a Quantum Key over a QBNI with n trustful nodes in Q3P

text which can be reused as a key material, it is delivered over $n - 1$ nodes. We can map an arbitrary message to the plaintext i.e. we can transmit the plaintext itself over the QBNI. However, we map a quantum random key to the $X_{0,3}$ such that the initial plaintext will be encrypted with $X_{0,3}$ after the transmission. To avoid ambiguities, $X_{0,3}$ denotes a plaintext, because it is not necessarily a quantum key but for the implementation it is useful to use a quantum key because there can exist untrusted nodes in the graph.

To show the concepts, every node on a path is assumed to be reliable in Fig. 5.5. Hence the transmission of a (quantum) random secret key $X_{0,3}$, denoted by `(Q)URAND()` from node N_0 to node N_3 will succeed. The computation steps for the transmission of a key over a QBNI is given in Alg. 5.1. In the first two steps of a QKD multihop key transport, the root and goal backbones need to be specified. The rootID denotes Alice' nodeID, whereas the goalID denotes Bob's nodeID on a path in the graph from Alice to Bob. As the encryption function we choose the logical `XOR` operation which is reversible to use the information theoretically secure OTP. Hence if we apply an arbitrary key K on a plaintext X twice we derive the plaintext as well.

An example of a valid hop by hop transmission over a trusted QBNI is given in Fig. 5.6. Transmitted is a sequence of two nibbles. The rootID=0 and the goalID=3 are given, hence Alice generates a true (quantum) random number $X_{0,3}$ with a sufficient entropy i.e. shannon entropy. Hence a true random number will be isomorph, similar to quantum key material (QKM). The 2 adjacent nodes (N_0, N_1) generate a Quantum Key $K_{0,1}$. The node N_0 encrypts $X_{0,3}$ as follows

$i = 0 : Y_{0,1} = (X_{0,3} \oplus K_{0,1})$ and transmits the ciphertext $Y_{0,1}$ to node N_1. After receiving $Y_{0,1}$, the node N_1 decrypts the ciphertext with the decryption function and derives $i = 0 : X_{0,3} = (Y_{0,1} \oplus K_{0,1})$. In the next step, the 2 adjacent nodes (N_1, N_2) generate a quantum key $K_{1,2}$, the node N_1 encrypts $X_{0,3}$ as follows $i = 1 : Y_{1,2} = (X_{0,3} \oplus K_{1,2})$ and transmits the ciphertext $Y_{1,2}$ to node N_2. Now the node N_2 decrypts the ciphertext with the decryption function and obtains $i = 1 : X_{0,3} = (Y_{1,2} \oplus K_{1,2})$. All nodes proceed iteratively as mentioned before until Node N_3 obtains the key $X_{0,3}$. Once the hop by hop mechanism is finished, Alice and Bob use $X_{0,3}$ as a One-Time-Pad to encrypt and decrypt the message to transmit. Hence every root node has knowledge about the following components of the transmission

$$N_j = \{X_{j,n-1}, K_{i,i+1}, Y_{i,i+1}\} \tag{5.3}$$

and every intermediate node consists of

$$N_i = \{K_{i-1,i}, Y_{i-1,i}, X_{j,n-1}, K_{i,i+1}, Y_{i,i+1}\} \tag{5.4}$$

whereas every goal node consists of

$$N_{n-1} = \{K_{n-2,n-1}, Y_{n-2,n-1}, X_{j,n-1}\} \tag{5.5}$$

and leads to the fact, that every reliable, intermediate node knows the initial random key $X_{j,n-1}$ i.e. $K_{0,3}$. The concept of the reliable nodes also allows us to transmit the plaintext instead of the

```
Data: A graph, Quantum Key, rootID, goalID
Result: Transmission of a a Quantum Key over a QBNI
1 Initialization: $rootID = i = j$, $goalID = n - 1$;
2 Generate Random Key $X_{j,n-1}$;
3 while $i < n - 1$ do
4     Generate Quantum Key $K_{i,i+1}$ between the two adjacent nodes $i, i + 1$;
5     Node $i$ encrypts $Y_{i,i+1} = X_{j,n-1} \oplus K_{i,i+1}$;
6     Node $i$ sends $Y_{i,i+1}$ to adjacent node $i + 1$ over an arbitrary channel;
7     Node $i + 1$ decrypts $X_{j,n-1} = Y_{i,i+1} \oplus K_{i,i+1}$;
8     $i = i + 1$ ;
9 end
```

Algorithm 5.1: Quantum Backbone Network Interconnection of the Transmission of a Quantum Key over n nodes

quantum random key $X_{0,3}$, generated by Alice. However, it is useful to transmit a seperated key $X_{0,3}$ hop by hop, instead of the plaintext if we have $k < n$, unknown and untrusted intermediate nodes on a path.

Figure 5.7 illustrates the associations between a quantum node a QKD-link and a QKD-stack to implement an unconditionally secure QBNI with more than one path. In general, this figure shows the architecture of quantum networks, but is not restricted to. The QBNI consists of QKD link devices made of quantum routers (QR), WDM components, Faraday-Michelson interferometer (FMI) and QKD protocols, to establish a quantum connection between two adjacent nodes.

							N_2			
				N_1						
		N_0							N_3	
	$X_{0,3}$	$K_{0,1}$	$Y_{0,1}$	$X_{0,3}$	$K_{1,2}$	$Y_{1,2}$	$X_{0,3}$	$K_{2,3}$	$Y_{2,3}$	$X_{0,3}$
0	0	1	1	0	0	0	0	1	1	0
1	1	0	1	1	0	1	1	1	0	1
2	0	1	1	0	0	0	0	1	1	0
3	1	1	0	1	0	1	1	0	1	1
4	0	1	1	0	1	1	0	0	0	0
5	0	1	1	0	0	0	0	1	1	0
6	1	0	1	1	0	1	1	0	1	1
7	1	1	0	1	1	0	1	0	1	1
$(x_i, y_i)i$:	0	0		1	1		2	2		
$f(x_i, y_i)i$:			0			1			2	
$(a_j, b_j)j$:		0	0		1	1		2	2	
$f^{-1}(a_j, b_j)j$:				0			1			2

Figure 5.6: Transmission of a Quantum Key over a Quantum Network

A QBNI consists of a set of nodes, as shown before. The nodes in the graph represent quantum nodes and every node is connected to at least one quantum node in the QBNI. Every quantm node has at least one quantum link to connect two nodes, respectively and every link can consist of arbitrary QKD stacks, greater than 1. QKD is restricted in the distance and the key generation rate because of the losses on the noisy quantum channel until quantum repeaters are used to expand these limits by a secure amplification of qubits without decreasing the security level. By the use of concatenation of several short distance end-to-end QKD links, in combination with secure multi hop propagation protocols, the end to end distance of two communication parties increases. Furthermore, every QBNI can consist of n QKD subnets. Figure 5.8 shows a QBNI, denoted as a graph $G = (V, E)$, with vertices $v \in V$ and edges $e \in E$. The structure of the QBNI is based on redundancy point to point connections. As we will show later, these redundancies enable unconditional secure key distribution over the network even if there are unreliable intermediate nodes in the graph. Alice and Bob are authenticated on the quantum channel per definition. Every node in the graph represents a single QKD node in the QBNI which consists of at least one QKD Link device. For simplification, the structure of a single QKD node is shown in Fig. 5.9. It shows the concept of the experiments. The quantum point to point protocol (Q3P) engine provides cryptographic functions and handles the exchange of qubits over the quantum channel and the classic bit exchange over the public channel. The generated quantum keys are handed out over textfiles and the standard output. The edges from Fig. 5.8 represent

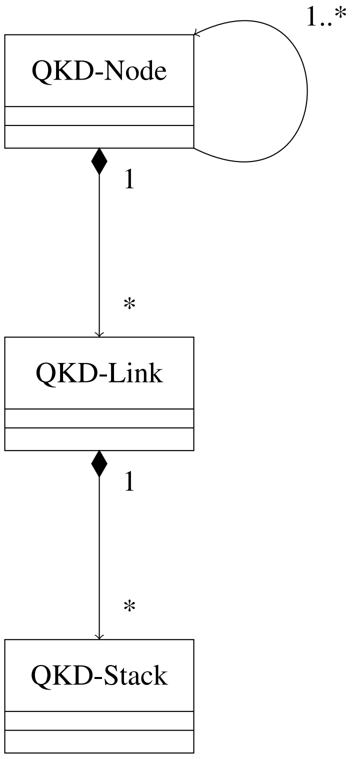


Figure 5.7: UML of the Node, Link and Stack associations in a QBNI

the WDM quantum channels between two adjacent nodes in the graph. By construction of the QBNI one can see that two distant nodes are connected via at least one quantum channel and a public channel and there exist more than one paths from A to B. The public channel is used to enable the postprocessing methods after deriving a raw key by the quantum infrastructure between two nodes. A postprocessing method is the error estimation or correction and this is done in a classical way, for instance. This extension is shown in Fig. 5.10, where one untrusted (U) and another trusted (T) path are shown. A path in the graph is an untrusted path if there exists at least one untrusted, intermediate node and a path is a trusted path, otherwise. The set of untrusted nodes is denoted by $\mathcal{U} = \{N_1, N_2\}$ and the set of trusted nodes in the graph is denoted by $\mathcal{T} = \{N_0, N_3, N_4, N_5\}$. Alice and Bob are trusted per definition because they are authenticated and there exists at least one trusted path in the graph from N_0 to N_3.

The propagation is similar to a transmission over a single trusted path. The iterations from Alg. 5.1 are done from Node N_0 to Node N_3 over the set of trusted, intermediate nodes $\mathcal{T}_{int} = \{N_4, N_5\}$. This results in a random key $X^T_{0,3}$. Despite the fact that the alternative path over the untrusted, intermediate nodes $\mathcal{U}_{int} = (N_1, N_2)$ is untrusted, the propagation will be done as shown before, hence the path is assumed to be reliable for the calculation of the keys,

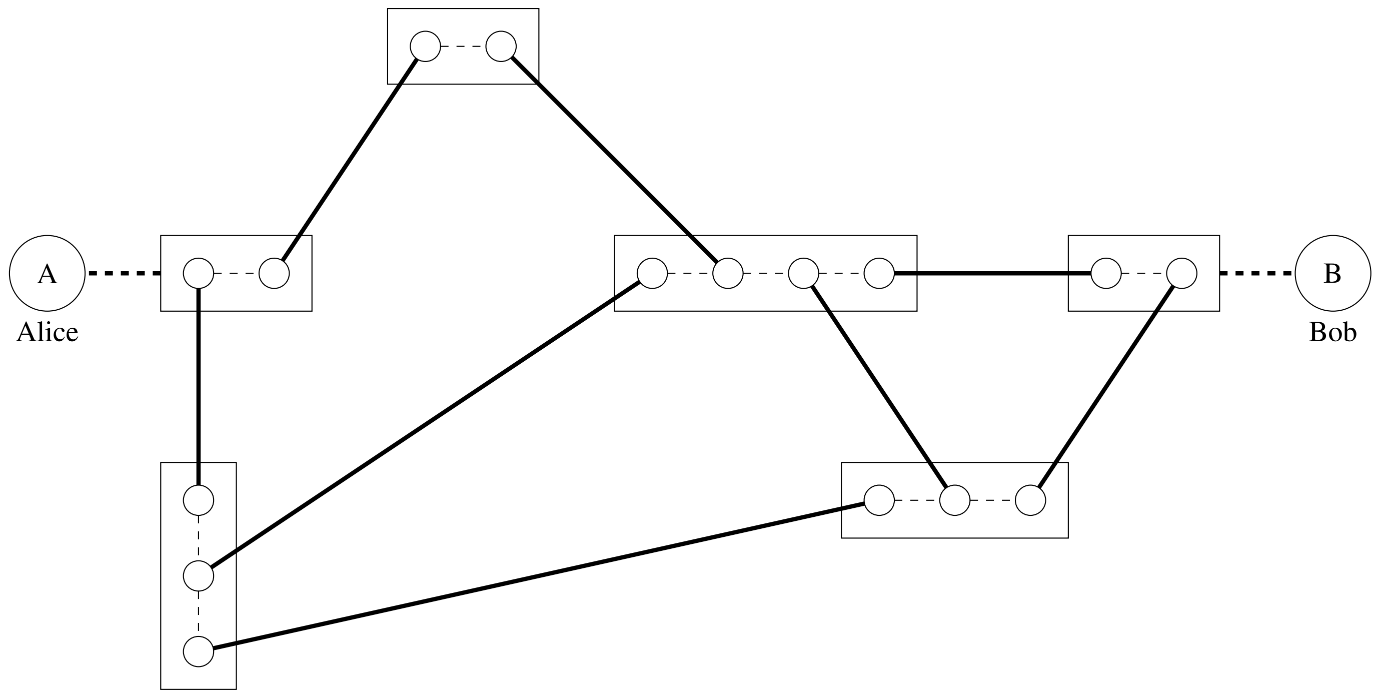


Figure 5.8: Quantum Backbone Network Interconnection Model

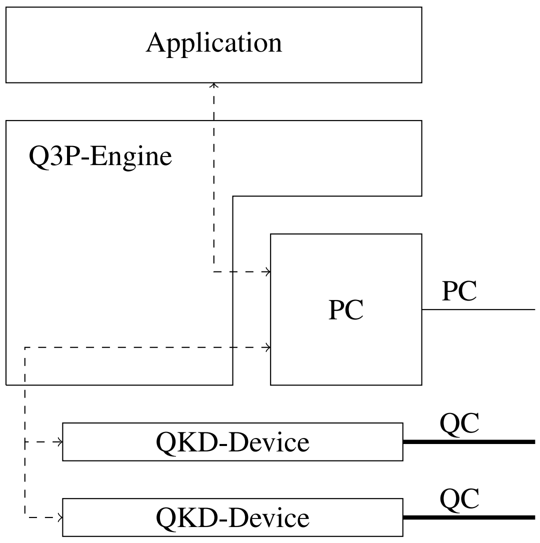


Figure 5.9: Network Node Structure of a QBNI in Q3P

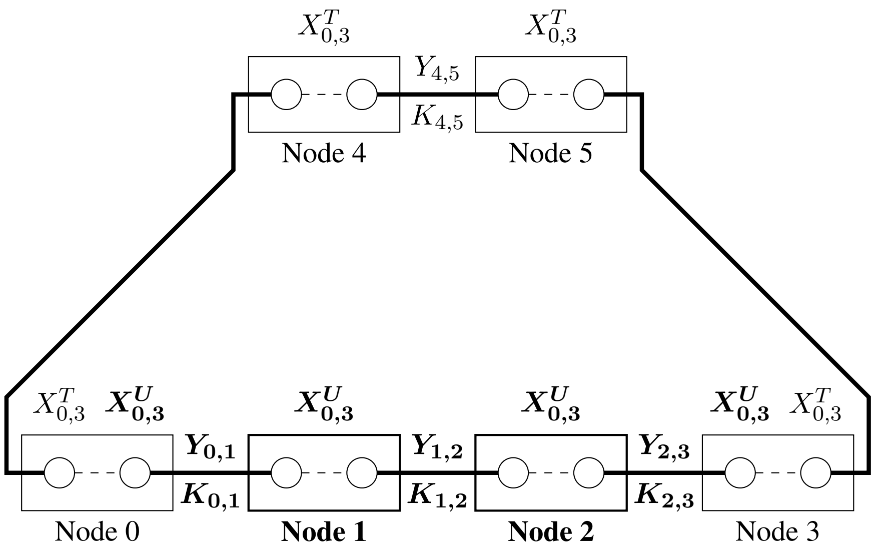


Figure 5.10: Transmission of a Quantum Key over a QBNI with $k = 2 < n$ untrusted nodes - $K_{final} = \left(X^U_{0,3} \oplus X^T_{0,3}\right)$

respectively. The untrusted computation results in the untrusted key $X^U_{0,3}$. However, Alice and Bob hold 2 keys and one out of these keys is a random, unconditionally secure quantum key. As a result, both compute $K_{final} = \left(X^U_{0,3} \oplus X^T_{0,3}\right)$. The superscript R_j specify if a path is reliable (trustful) or unreliable i.e. it is mapped to “1” if the corresponding path is reliable and it is mapped to “0” otherwise. A disjunction of R_j specifies if a quantum random key, transmitted

$$I\left(R_j\right) = \begin{cases} T, & \text{iff } b = 1 \\ U, & \text{iff } b = 0 \end{cases}$$

R_0	R_1	$R_0 \vee R_1$
0	0	0
0	1	1
1	0	1
1	1	1

Figure 5.11: Final Key Superscript for the Multihop Mechanism over 2 paths in a QBNI

over the QBNI is trustful or not. This is shown in the following equation.

$$R = \bigvee_{0 \le j \le m-1} R_j \tag{5.6}$$

The composition of a quantum key over a network with $k < n$ untrusted, intermediate nodes and $m-1$ paths is denoted by K^R_{final} and specified by the modulo 2 arithmetic by

$$K^R_{final} = K^{R_0}_{i,n-1} \oplus K^{R_1}_{i,n-1} \oplus \ldots \oplus K^{R_{m-1}}_{i,n-1} \tag{5.7}$$

and an instance is shown in Fig. 5.11. Hence a trustful quantum key demands the existence of at least one trustful path in the QBNI. Furthermore

$$\mathcal{T} \cap \mathcal{U} = \emptyset \tag{5.8}$$

holds i.e. if a node is unknown or untrusted it can not be interpreted as a reliable one. Figure 5.12 shows a layer model for the quantum point to point protocol for the integration into IPv4/IPv6. IPv6 is interpreted as a standard application, where Q3P provides quantum key material instead of the Diffie Hellman key exchange. On the Quantum Layer, Q3P establishes a communication over an authenticated, noisy quantum channel, to transmit the qubits from Alice to Bob. The network layer provides public IPv4/IPv6 communication, whereas the key postprocessing layer implements an intrinsic use of Q3P.

Intrinsic means, that Q3P needs itself to administrate. Quantum key material is used to encrypt and authenticate the messages on the Key Postprocessing Layer. Q3P does this via `q3p_send` and `q3p_recv` commands. If an application requires quantum key material it is delivered over Q3P via the mentioned commands. By definition, Q3P is an intrinsic protocol and it interprets every module as an application that demands quantum key material but Q3P creates the quantum key material by itself. The invocation of `q3p_send` with the corresponding parameters allows Alice to send key material to Bob. The parameters are the channel number e.g. 1 and the options that the message should be transmitted authenticated and encrypted and the key material as a plaintext. Q3P provides the authentication methods instantly or delayed over an authentication channel. However, the plaintext key material gets encrypted via Q3P. The key is specified by a character string and the keysize is specified by a 32 bit integer. To receive the key material it is necessary to invoke the `q3p_recv` command on Bobs side. The initialization

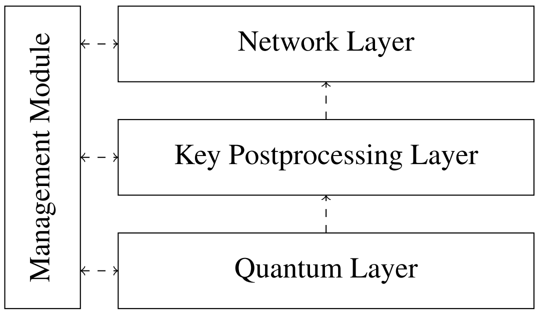


Figure 5.12: Layer Structure of Q3P

of the Q3P system is based on the universally unique identifiers from the virtual systems. How one can find out the corresponding uuid is described in Appendix B.

The qubits are transmitted between Alice and Bob over the Quantum Channel. However, during a transmission there exists an interaction between the Quantum Channel, the physical implementation, the environment and an eavesdropper. In summary, this is known as noise and changes the value of a qubit during the transmission. Hence a measurement with the correct basis on Bob's side $\left(K^{'}_{sifted}\right)$ results in a divergent outcome than the original value on Alice's side (K_{sifted}), both with size of n bits. It follows that the key on Alice's side does not match that on Bob's side. Furthermore Alice and Bob initiate the reconcilation phase, or former known as error correction stage, to reconcile the divergent random bit streams. The key procedure in the Cascade error correction and estimation is a binary search, denoted as `BINARY`. `BINARY` calculates from every Input Stream $A_{i,j}$ at most two substreams $A_{i+1,1}, A_{i+1,2} \subseteq A_{ij}$, where i means the iteration and $j \in \{1,2\}$ denotes the first or second half of the original stream $A_{i,j}$. However, the number of bits does not have to be necessarily equal in both substreams. Furthermore, to define a rule to calculate the substreams, the length of the first substream is specified by

$$n_{i+1,1} = \texttt{SIZE\_OF}\Big(A_{i+1,1} = \Big\{x_0, \ldots, x_{\lceil \frac{n-1}{2} \rceil - 1}\Big\}\Big) \tag{5.9}$$

which is at least as long as the length of the second substream, which is defined by

$$n_{i+1,2} = \texttt{SIZE\_OF}\Big(A_{i+1,2} = \Big\{x_{\lceil \frac{n-1}{2} \rceil}, \ldots, a_{n-1}\Big\}\Big) \tag{5.10}$$

and at most $n_{i+1,2} + 1$. Alice and Bob perform the binary search over the public channel for an arbitrary substream of the sifted key. Both of them calculate the parity bits from their streams for every iteration. In the first iteration, Alice calculates the parity from the entire string, as Bob does. Bob checks the parity bits for equality. If this check assigns to *true*, the number of errors in the substream is even $(2k,\ k \in \mathbb{N}_0)$. However, if the check assigns to *false*, the number of errors in the substream is odd $(2k+1,\ k \in \mathbb{N}_0)$. Hence if the number of errors in the stream is odd, Alice and Bob calculate the parity of the first half of the stream. Bob calculates, whether the odd errors occured in the first or in the second half of the stream by comparing $p_B\left(A_{i,j}\right)$ with the parity bit from Alice $p_A\left(A_{i,j}\right)$. Hence Bob concludes the half of the stream, and set $j \in \{1,2\}$. After a couple of iteration rounds, the erroneous bit is found. It will be corrected

Error Correction Technique	Q3P Implementation
Cascade Error Correction	✓
Hamming Error Correction Code	
Linear Error Correction	
Low Density Parity Check	✓

Table 5.1: Overview of the Error Correction in QKD Protocols

by the XOR modulo 2 arithmetic $b_i^{'} = (b_i + 1) mod 2$. However, Alice has the original, infallibly key, the modulo 2 arithmetic will be denoted as BITFLIP($B_{i,j}$), because the errors occur only on Bob's side. At this stage, the number of bits in the substream is 1. Hence the bitflip will succeed. Algorithm 5.2 shows the computation steps. An example of using cascade error correction algorithm will be shown in Fig. 5.13, page 86.

```
Data: Two Strings $X_{i,j} = \{x_0, \dots, x_{n-1}\}$ s.t.
      $x \in \{a, b\},\ X \in \{A, B\} i,\ n \in \{\mathbb{N}_0\},\ j \in \{1, 2\}$
Result: Error Correction of $k$ erroneous bits with an odd number of errors
1  Initialization: $i = j = 0$, $X_{ij}$;
2  Bitflip $k$ erroneous Bits;
3  while $ERRORNUMBER > 0$ do
4      $n$ =SIZE_OF($X_{ij}$);
5      while $n > 1$ do
6          $i = i + 1, j = 1$;
7          $X_{ij} = \left\{x_0, \dots, x_{\lceil \frac{n-1}{2} \rceil - 1}\right\}$;
8          CALC_PARITY($X_{ij}$);
9          if $(p_A(A_{i,j}) = p_B(B_{i,j}))$ then
10             $j = 2$;
11             $X_{ij} = \left\{x_{\lceil \frac{n-1}{2} \rceil}, \dots, a_{n-1}\right\}$;
12         else
13             skip;
14         end
15         $n$ =SIZE_OF($X_{ij}$);
16         return($X_{ij}$);
17     end
18     Assert($n = 1$);
19     BITFLIP($B_{ij}$);
20     PERMUTATION($X_{ij}$)
21 end
```

Algorithm 5.2: Cascade Error Correction

The LDPC consumes 96 bit and one hash tag per transmitted bit during the correction of one single erroneous bit. The quantum key material is based on two 96 bit hash tags. The first hash tag is the result of pre-calculated hash tables with reusable hash tag instances, whereas

$$\underbrace{\overbrace{p(A_i)=p(B_i)}^{\pi_0}}_{b_i^{'}=(b_i+1)mod2}\quad \underbrace{\overbrace{p(A_i)=p(B_i)}^{\pi_i}}_{b_i^{'}}\quad \underbrace{\overbrace{p(A_i)=p(B_i)}^{\pi_i}}_{b_i^{'}}$$

0	1 1 1		0 0 0 0		**0** **1** 1 1 1 1 **0**
1	0 0 0		1 1 1 1		0 0 0 0 0
2	1 1 1		1 1 1 1		0 0 0 0
3	0 0 0		0 0 0 0		1 1 1 1
4	0 0 0		**0** **1** 1	1 1 **0**	0 0 0
5	0 0 0		0 0 0	0	1 1 1
6	1 1 1		1 1 1		1 1 1
7	1 1 1		1 1 1		0 0 0
8	1 1	1 1 1	1 1		1 1
9	**1** **0**	0 0 **1**	1 1		1 1
10	**0** **1**	1	0 0		0 0
11	**0** **1**	1	0 0		1 1
12	0 0		**0** **1**		0 0
13	1 1		1 1		1 1
14	0 0		1 1		1 1
15	1 1		0 0		0 0
$p(A_i) = p(B_i)$	f	$t\ f\ f\ t$		$f\ t\ f\ f$	$f\ f\ f\ f$
Stream		2 1 1 2		1 2 1 1	1 1 1 1

Figure 5.13: Cascade Error Estimation and Correction in Q3P

the second is used for polynomial multiplication for the final hash tag. That means at least 192 bit are required for authentication in the quantum network. That means it is recommended to use the Cascade error correction algorithm in the implementation.

5.3 Quantum Point to Point Protocol - Q3P

A communications protocol that uses a set of single point to point infrastructures is the Quantum Point to Point Protocol or formally known as Q3P. Its design is similar to a client server architecture. A Q3P link interpretes all adjacent nodes as a potential QKD device and distinguishes to different kinds of nodes, a QKD device and a user centric application. The first pushes the quantum key material into the common key store, whereas the latter uses the communication facilities, provided by the Q3P link. By the integration of quantum infrastructure, from classic gateway to gateway communication follows quantum point to point communication. The quantum point to point protocol allows two distant nodes Alice and Bob, to communicate unconditionally secure and unconditionally authenticated, over wide areas within a QBNI. By contrast to classic communication protocols, which are based on the mentioned OSI or DoD layer protocol stack, Q3P is a protocol of intrinsic structure. Q3P is based on modules, which represent applications who perform several tasks and it allows to deliver quantum key material over a QBNI secure. Hence Q3P uses its own key material for interprocess communication between two adjacent modules to synchronize the protocol stages over unsecure channels. So the classical procedures such as key sifting on Alice's side, will be synchronized with the stages from key sifting on Bobs side via `q3p_send` and `q3p_recv`. That means every classical transmission is unconditionally authenticated and secure transmitted over the public channel. The model of the experiments is shown below. The communication between the quantum infrastructure and the Linksys router is secure per definition. The quantum channel and the first public channel are used to administrate the quantum key material and enable secure communication via the Q3P commands. The queue provides the key material and delivers it to the internet protocol instead of the diffie hellman key exchange. Hence the second public channel is used to provide the IPv6 connection between two distant nodes. The key store provides quantum keys to Q3P and enables unconditionally secure communication on the public channel to administrate the key material. Because it is not possible to extract the quantum keys directly from Q3P it is necessary to combine a quantum random number or a true random number with the quantum keys from Q3P. The random number is delivered from one endpoint to another and can be used to establish an unconditionally secure communication. It is used as a one time pad.

QKD Stack

The QKD protocol stack performs postprocessing methods on QKD key material such as sifting, error correction and privacy amplification. Privacy amplification and other preprocessing techniques are not in the scope of this work. The postprocessing methods are implemented as a classical UNIX pipeline over the standard outputs. If we use Q3P to administrate the internal communications on the local machines, which are assumed to be secure, the key generation decreases. The information is redirected from one module to the next module and the stages of the sifting phase by Alice will be synchronized with the stages from Bob by the use of `q3p_send` and `q3p_recv` commands.

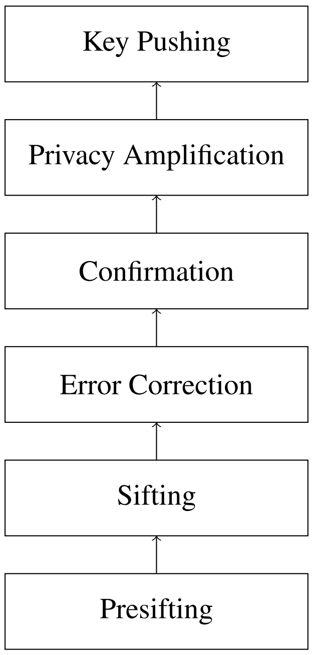


Figure 5.14: Protocol Stack of Quantum Key Distribution

The task of the Q3P protocol stack is to generate final quantum key material for Q3P interprocess communication (IPC) over unsecure channels. Presifting prepares the raw key material for the subsequent sifting phase. It provides timetag information for the successive sifting module, `q3p_si` from the time taking machines (TTM) and the dedicated hardware. The TTM are responsible for the time synchronization to administrate the modules and the corresponding tasks. The sifting module implements `BB84_Sifting` by manipulating quantum tables with dedicated detector click information. The computation rounds were explained in Ch. 4. Two different methods of error correction are present in QKD environments, which are also referred to as reconsiliation techniques. For noisy channels, as QCs are, the low density parity check (LDPC) [15] is a method for reconciliation by the use of bipartite graphs with two disjoint sets of nodes, such that every node from the set of nodes is connected with one vertex from the set of vertices. Cascade error correction is another present reconsiliation method in QKD environments. By logical `AND` operation, a random BIGINT results in a bitwise parity check. Privacy Amplification is used to be designed by Toeplitz matrices. A Toeplitz matrix is an $n \times m$ matrix where $A_{i,j} = A_{i+1,j+1} = a_{i-j}$ holds. It is named by the German mathematician Otto Toeplitz

(* 1881 – † 1940). A Toeplitz matrix is specified as follows.

$$A = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-n+1} \\ a_1 & a_0 & a_{-1} & \ddots & \ddots & \vdots \\ a_2 & a_1 & a_0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & \ddots & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix} \tag{5.11}$$

Key pushing forwards the final quantum key material to the key store and notifies the system of a new, final quantum key material in the common key store. The key material from the key store can be used to encrypt and decrypt Q3P messages to transmit unconditionally authenticated and unconditionally secure encrypted.

QKD Pipeline

Q3P has a modular construction. Because of its intrinsic structure, Q3P uses its own quantum key material (QKM) to synchronize processing stages between the distant nodes in the QBNI. One single module processes only one specific task. Hence it is possible to add or remove modules and modify the workflow to optimize it corresponding to the dedicated task. Every QKD Module needs a Q3P service interface. As a result, all modules are able to communicate over unsecure channels by the use of `q3p_send` and `q3p_recv`. These commands use the final key material from the common key store to encrypt and authenticate messages unconditionally secure.

QKD Network Management

The Q3P network management uses final key material, created by the Q3P protocol stack and performs the transport over the public channel. It executes key storage methods and provides final key material for IPC. The key store provides final key material for `q3p_send` and `q3p_recv` commands whereas the crypto engine realizes the cryptographic algorithms i.e. encryption and decryption by the `XOR` operation or verifying if the length of the key is identically to the length of the message.

Q3P Message Queue

The message queue is a device or an application as any other QKD devices to Q3P. The message queue is responsible to fetch the uuid for a peer and the queue. It requests the Q3P link for device pairing that it is connected secure to the link device. A random source delivers bits on Alice's side. If Bob receives a quantum key it replies with an authenticated and encrypted acknowledgment and pushes the quantum key into his local message queue. Alice fetches the acknowledgment and pushes a copy of the random key into her message queue. Hence the key is ready for further postprocessing methods.

Key Store

The Q3P key store is used by a Q3P link and it provides quantum key material for internal IPC over the Q3P commands `q3p_send` and `q3p_recv` to send and receive a message on the public channel. It is similar to a library and it is based on the Berkeley system data base. The common key store stores the quantum key material and provides it for the module.

Crypto Engine

The crypto engine is a module that provides encryption and authentication in Q3P. It checks if the lengt of the plaintext to be transmitted fits the length of the quantum key used. It provides cryptographic functions such as the `XOR` operation on the key and the plaintext to encrypt the plaintext by the use of a one time pad. The evaluation hash tags are provided by the crypto engine. The length of an instance of a hash tag is 32, 64, 96 or 128 bit. If delayed authentication is necessary, it has to be triggered externally because of the trade-off based on the Cascade error correction algorithm.

CHAPTER 6

Experimental Evaluation

Quantum key distribution is a promising method to deal with future demands on communications security. It enables unconditional security for short distances lower than 100 km to transmit light quantas reliable. However, if we combine QKD with classical solutions, we can bridge longer distances and achieve communication over the internet between two distant nodes. The key generation rate decreases with increasing distance because the transmission losses increase. The security level depends on the applied technique. If we use classical solutions such as the Diffie Hellman key agreement protocol we can bridge longer distances, but its security is based on unproven computational complexity assumptions. If we use QKD to secure IPv6 we need intermediate nodes between the communication partners because of the properties of quantum channels and their behavior over larger distances. However, QKD provides unconditional security, despite the fact that we have a set of unknown or unreliable intermediate nodes between the endpoints. The system has been restarted after every instance of a measurement cycle.

6.1 Experimental Setup

The measuring setup of the experiments taken, is based on the virtualbox virtualization environment and a set of virtual machines where every machine has its own network interface. It runs on a physical machine and virtualizes the setup environment in an isolated network. As we have shown in the sections above, a QBNI consists of a set of intermediate nodes (Fig. 5.1) because of the properties of quantum channels to establish a communication between two distant nodes in the quantum network. The QBNI uses the properties of the quantum channels via wave division multiplexing to multiplex and switch between the public channel and the authenticated quantum channel. Because the public channel competes with the quantum channel over WDM we use the new public channel between the two routers to enhance the transmission speed and balance the load on the quantum network to discharge the QBNI and improve its efficiency. Two virtual routers are integrated as virtual machines to establish a peer to peer connection between the endpoints. By Q3P's point of view every module is an application. Hence we interprete the IPv6 protocol as a user centric application and integrate it as shown in Fig. 6.1. Time synchroniza-

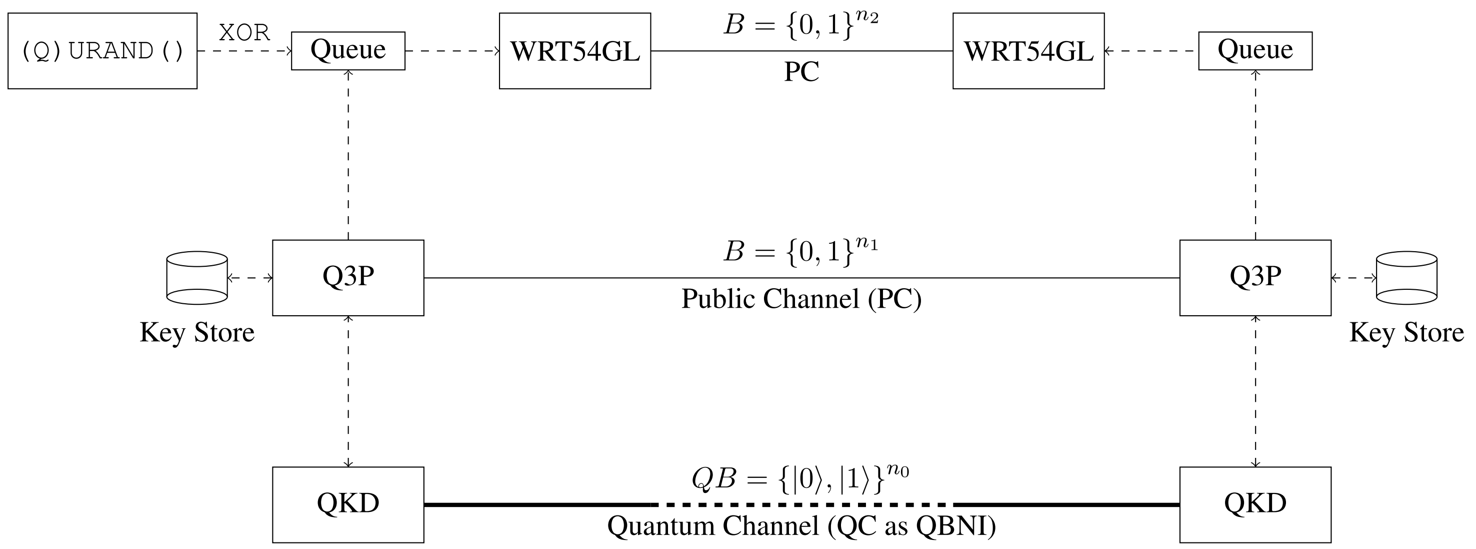


Figure 6.1: QKD Extension for IPv6 - $n_2 \leq n_1 \leq n_0$

tion in Q3P is done via time taking machines (TTM) over the management module. However, the virtual machines are synchronized because they run on one physical machine and the same clock is used on it. More information on Q3P and its architecture can be found by contacting the Austrian Institute of Technology GmbH (AIT).

6.2 Latency

IPv6 is the dedicated future solution for internet communications and it is intended to replace the IPv4 protocol. A lot of parameters influence decisions whether a technology is used in industrial applications or not. The latency is a factor that influences the use of a technical solution in a widely used protocol and the implementation into existing communication environments. It is the time delay in a communications system which depends on the system infrastructure and its physical implementation. That means the latency is the amount of time for a packet sent. At the beginning of this work we said, that two bottlenecks on QKD are the key generation rate and the distance between two nodes. The performance comparisons are done with an optimized router configuration on the WRT54GL. However, additionally application specific optimizations can be done for user centric applications such as audio, video, medical imaging or other dedicated tasks. But these optimizations are not in the scope of this work.

Quantum key distribution provides unconditionally secure key distribution between two communication endpoints over distances lower than 100 km. To show that QKD provides unconditionally secure key distribution between two distant nodes in a realistic environment and that QKD and IPv6 complement each other due to the modular design of IPv6 and Q3P, we compare the latency of the standard IPv4, IPv6 and the QKD system for two distant nodes.

Detailed results will be shown by non-parametric boxplots and probability density functions. The boxplot shows the sample minimum and maximum, the lower and upper quartile $(25\% - 75\%)$ and the median. It is illustrated by Fig. 6.2. The probability density function shows the probability for a continuous random variable to take on an instance of a value. The performance comparison shows that the QKD system is the slowest solution, yet. However, QKD is unconditionally secure and IPv4 and IPv6 are not because of the implemented security concepts such as the Diffie Hellman key agreement from IPv6. That means it depends on the task to fulfill which solution is required to use for securing communications.

IPv4 - Latency

IPv4 is the commonly known and widely used internet protocol version over wide areas. It is used in the range of non sensitive data transmission up to safety relevant communications [25] nowadays. We can use it for private banking, e-commerce or medical data processing. Figure 6.3 shows five independently taken latency measurements on IPv4 connections between two distant communication nodes. A numerical overview of the values of the IPv4 boxplots is shown in Fig. 6.4. The probability density function of the measurement results of Fig. 6.3 are shown in Fig. 6.5. It shows the probability that the transmission of an IPv4 packet is transmitted takes the time given on the x-axis.

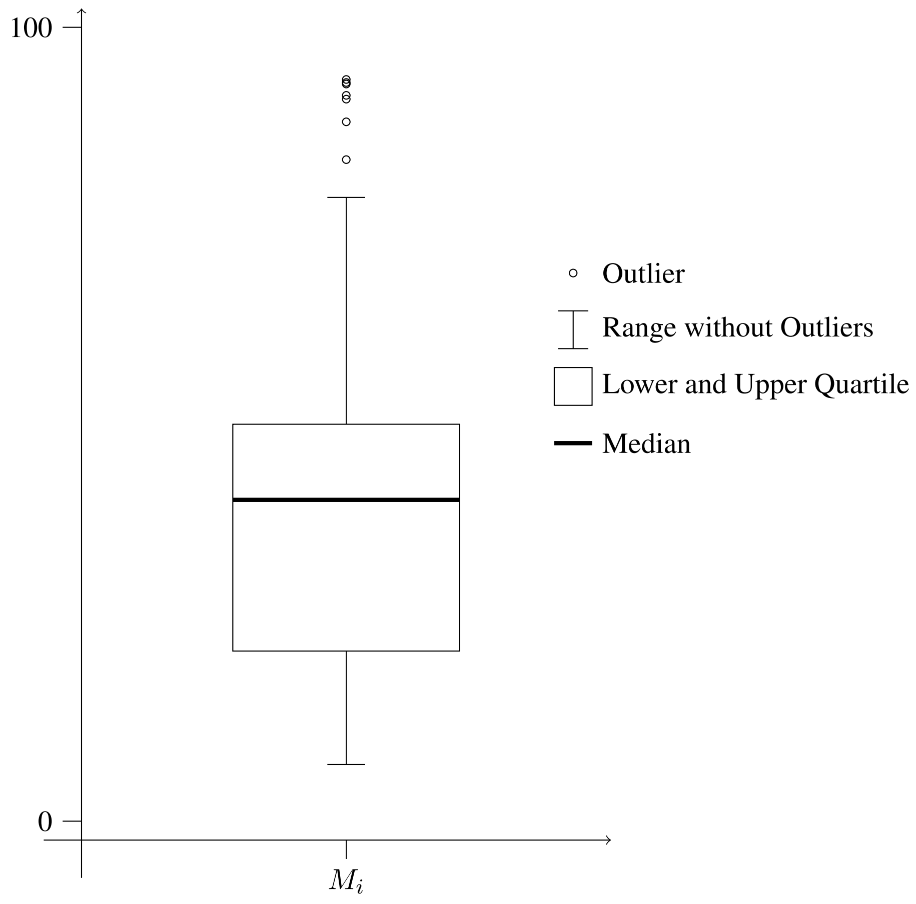


Figure 6.2: Boxplot Illustration of the Measurements M_i

IPv6 - Latency

The next generation internet protocol IPv6 enables end to end connectivity over large distances for 2^{128} devices. That means we can use unicast addresses to identify an interface on the network, uniquely. In the development of IPv6, care was taken to avoid conceptual weaknesses such as taken in IPv4. Figure 6.6 shows the boxplot of the measurement results for an IPv6 packet between two communication nodes. A numerical overview of the values of the IPv6 boxplots is shown in Fig. 6.7. When comparing the latency measurements one can see that IPv6 is somewhat faster than IPv4 in general. However, the disturbing influences that affect behavioral deviations from the arithmetic mean of the measured values are similar to IPv4. The disturbing influences are caused by the noise on the channel. Figure 6.8 shows the probability density function of IPv6 packet traffic. The performance comparison of the packet traffic shows that the performance increases from IPv4 to IPv6 because of the optimized packet structure despite the fact that the packet size increased from 32 bit up to 128 bit. Hence, we can neglect the increased packet size. That means that the latency decreased from IPv4 to IPv6. One can see that some-

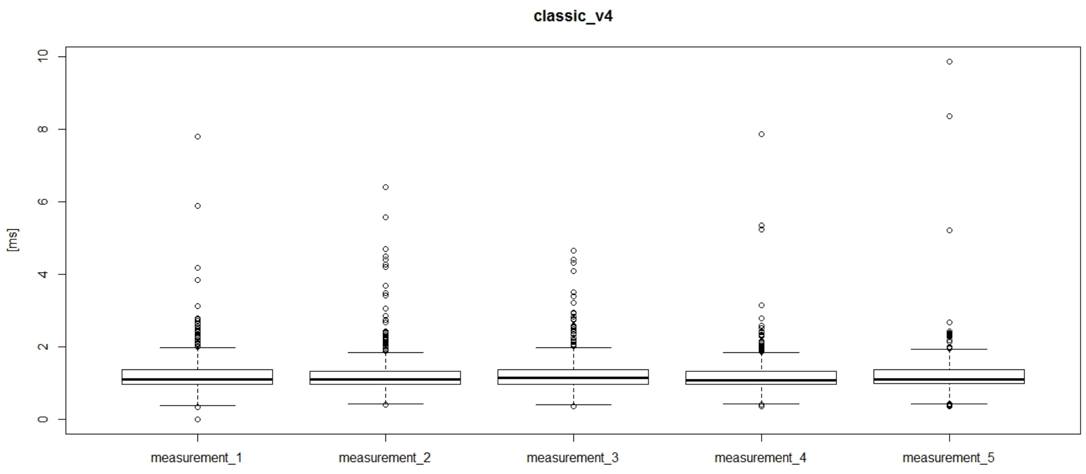


Figure 6.3: IPv4 Latency

IPv4 Latency [ms]	M_1	M_2	M_3	M_4	M_5
Lower Whisker	0.39	0.43	0.42	0.43	0.43
Lower Quartile	0.97	0.96	0.97	0.96	0.99
Median	1.11	1.11	1.14	1.09	0.99
Upper Quartile	1.38	1.32	1.38	1.32	1.37
Upper Whisker	1.98	1.85	1.97	1.85	1.93

Figure 6.4: Experimental Evaluation of IPv4

times IPv4 is faster than IPv6. However, the boxplot shows that the jitter in IPv6 is greater than the one in IPv4 and IPv6. However, Ipv6 is faster in general. Jitter specifies the deviation of the measurement values from the arithmetic mean. The probability density function of the IPv6 packets is similar to the one shown by the measurements for IPv4. The problem of the latency for IPv4 and IPv6 depends on the development of the computing power and the improvements on the network infrastructure, respectively. That means if we can reduce the absorption of optical fibres it is possible to increase the distance and speed of the packet transmission.

QKD - Latency

Quantum key distribution is a method to distribute key material between two communication parties unconditionally secure based on the properties of quantum physics. It can replace the Diffie Hellman key agreement protocol, used in IPv6 that two communication endpoints agree on a shared secret key and use it as a one time pad. The latency from IPv4 to IPv6 decreases but QKD is slower than IPv6 and IPv4, respectively. QKD provides the distribution of raw key material over a quantum channel and the raw key material is postprocessed with methods of

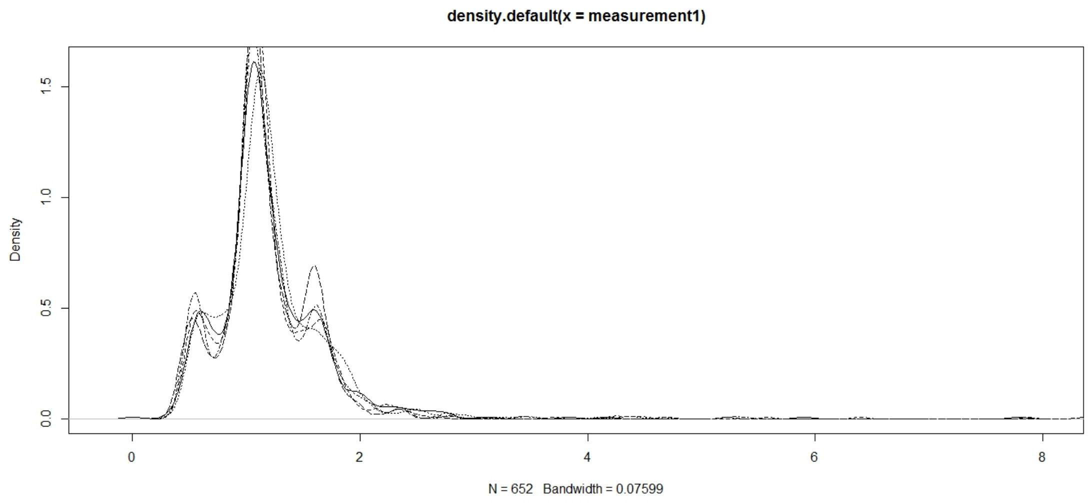


Figure 6.5: Latency Probability Density Function of IPv4

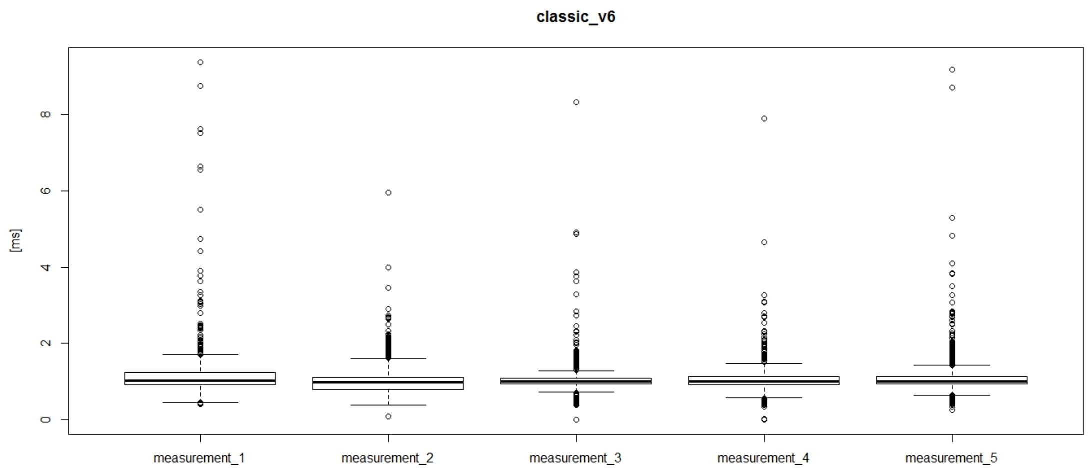


Figure 6.6: IPv6 Latency

classical computation over public channels to obtain a valid final key to secure communications. The latency of the QKD system is all over higher than the one in the previously mentioned classical IPv4 and IPv6 systems. Because quantum systems are not closed in real environments, there are interactions with the environment which influence the transmission of the states on the quantum channel. Tolerances of the infrastructure affect on the latency i.e. imperfections of the hardware implementation and differences in the quality of the detector environment result in an increasing latency and an increasing deviation of the measurement results on the QKD network. The quantum channel is a noisy channel i.e. the quantum channel is not a closed system. A

IPv6 Latency [ms]	M_1	M_2	M_3	M_4	M_5
Lower Whisker	0.46	0.39	0.73	0.58	0.64
Lower Quartile	0.92	0.79	0.94	0.92	0.93
Median	1.02	0.98	1.00	1.01	1.01
Upper Quartile	1.24	1.12	1.08	1.14	1.13
Upper Whisker	1.70	1.61	1.29	1.47	1.42

Figure 6.7: Experimental Evaluation of IPv6

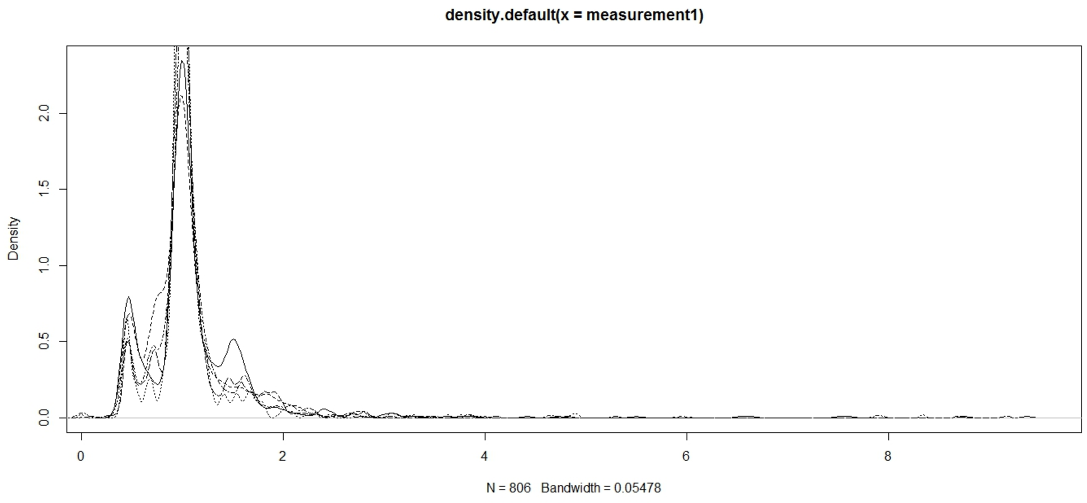


Figure 6.8: Latency Probability Density Function of IPv6

noisy channel is specified by the fact that the channel is memoryless i.e. if a sequence of actions is applied on the quantum channel, every instance is independent. Another property of a noisy channel is that noise acts locally. That means the noise of a quantum system does not correlate with the noise of a distant quantum system. Quantum states i.e. quantum superpositions can be disrupted by noise and physical imperfections of the implementation of the detectors and the detector efficiency influence the latency. Figure 6.9 shows the latency of a Q3P packet which enables unconditionally secure encryption and authentication. A numerical overview of the values shown in the QKD Boxplot is given in Fig. 6.10. It illustrates that the QKD system for unconditionally IPv6 security is slower than the classical IPv6 system based on the Diffie Hellman key exchange protocol. One can see that the range is greater than in IPv6 or in IPv4 with a classical packet transmission. The corresponding probability density function is shown in Fig. 6.11. It is different to the one shown by IPv4 and IPv6. The figures 6.9 and 6.11 illustrate the sensitivity of a quantum system and the transmitted states over the quantum channel against interferences from the environment and noise on the optical fibre.

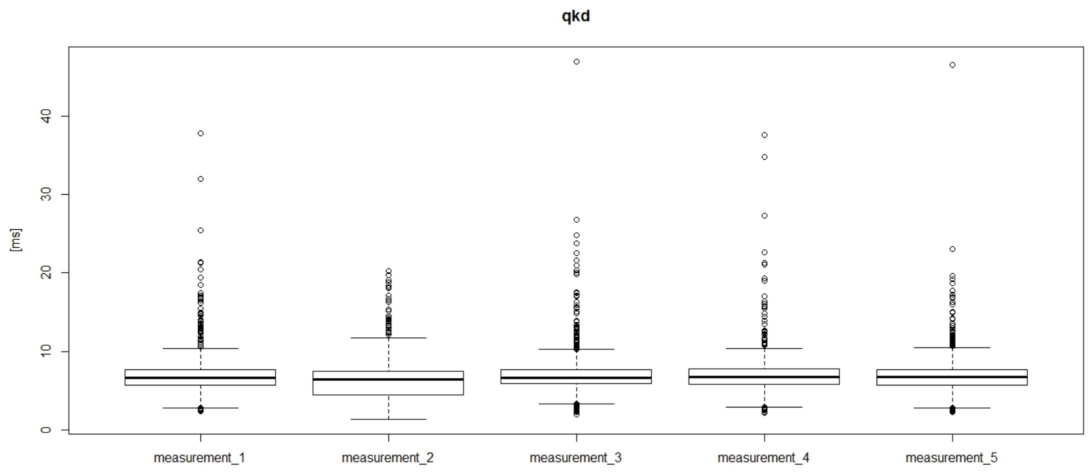


Figure 6.9: Quantum Key Distribution Latency (Q3P)

QKD Latency [ms]	M_1	M_2	M_3	M_4	M_5
Lower Whisker	2.82	1.31	3.25	2.87	2.78
Lower Quartile	5.70	4.44	5.88	5.82	5.71
Median	6.59	6.39	6.61	6.74	6.69
Upper Quartile	7.66	7.50	7.63	7.79	7.68
Upper Whisker	10.37	11.76	10.23	10.35	10.48

Figure 6.10: Experimental Evaluation of Quantum Key Distribution (Q3P)

6.3 Key Generation Rate

The key generation rate specifies the number of key bits generated per period. The key bits can be used as a one time pad to secure communications information theoretically secure. Because IPv6 and QKD perform end to end connectivity and the noise characteristics we have a look at the key generation rate for QKD systems.

QKD - Key Generation Rate

The key generation rate specifies the number of key bits generated per second. As we have shown in the previous section, a quantum network consists of QKD nodes which are connected via QKD links and every node can be connected with multiple QKD links. The redundant architecture of a QKD network to avoid only untrusted paths in the graph, results in an increased latency depending on the number of intermediate nodes on a single path. Consider a path with one intermediate node which generates raw key material. The raw key material needs to be post-processed via classical packet transmissions over the public channel. Hence we are interested in the link key generation rate of a single QKD point in the network. The boxplot of the link

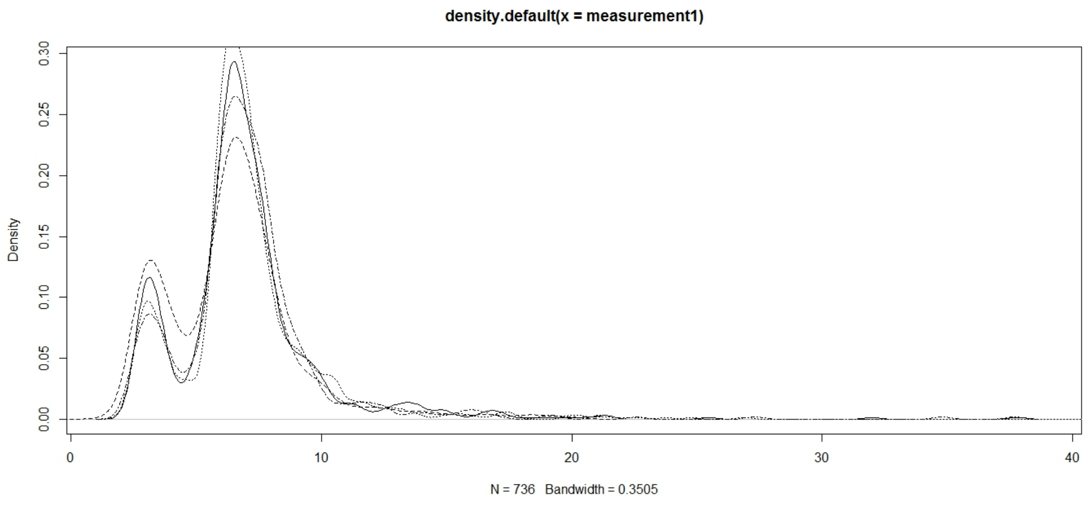


Figure 6.11: Latency Probability Density Function of Quantum Key Distribution (Q3P)

key generation rate is shown in Fig. 6.12. A numerical overview is given in Fig. 6.13. By con-

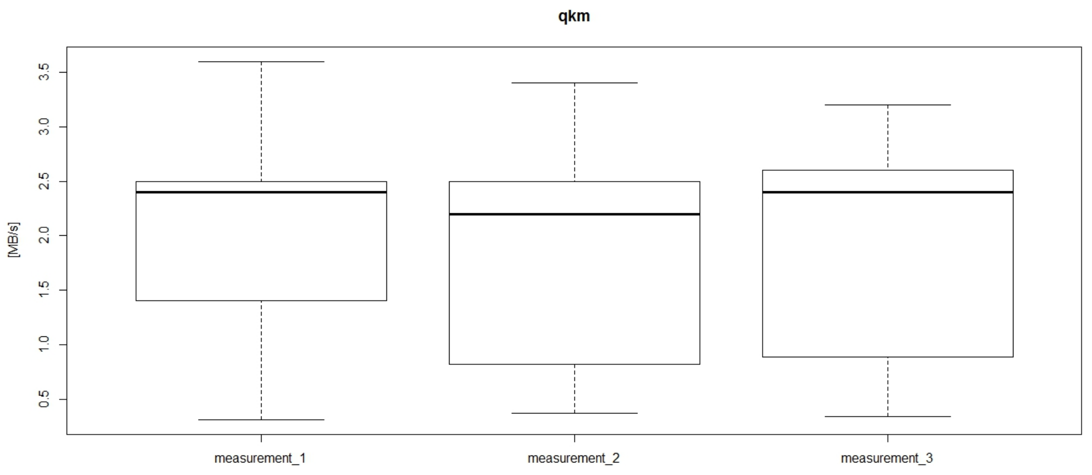


Figure 6.12: Keystore Key Generation Rate of Quantum Key Distribution (QKD)

trast to the boxplots from the latency measurements shown above one can see that the influences from noise on the quantum channel and the dependencies from external interferences decrease. This results in reduced jitter effects during the link key generation. Thus we see that the influences on the key generation rate are originally derived from the phyisical implementation of the quantum channel and the distance between two communication end points from the QBNI. The probability density function of the measurements is shown in Fig. 6.14.

Key Rate [Mbit/s]	M_1	M_2	M_3
Lower Whisker	0.31	0.37	0.34
Lower Quartile	1.40	0.82	0.88
Median	2.40	2.20	2.40
Upper Quartile	2.50	2.50	2.60
Upper Whisker	3.60	3.40	3.20

Figure 6.13: Experimental Evaluation of the Key Generation Rate of Quantum Key Distribution (Q3P)

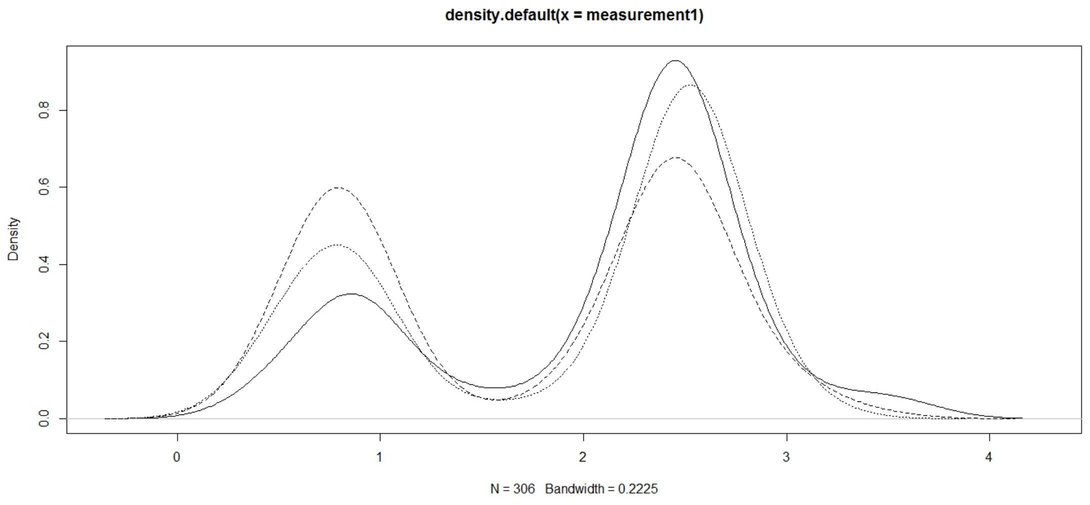


Figure 6.14: Probability Density Function of the Keystore Key Generation Rate

Key Generation Time

The key generation rate is a relative value which depends on the time. An absolute value is the key generation time. That means it specifies the time left to generate a quantum key in the network. The key generation rate and time are in coherence. Figure 6.15 illustrates boxplots of the key generation time for quantum key material in a QBNI. A numerical overview is given in Fig. 6.16.

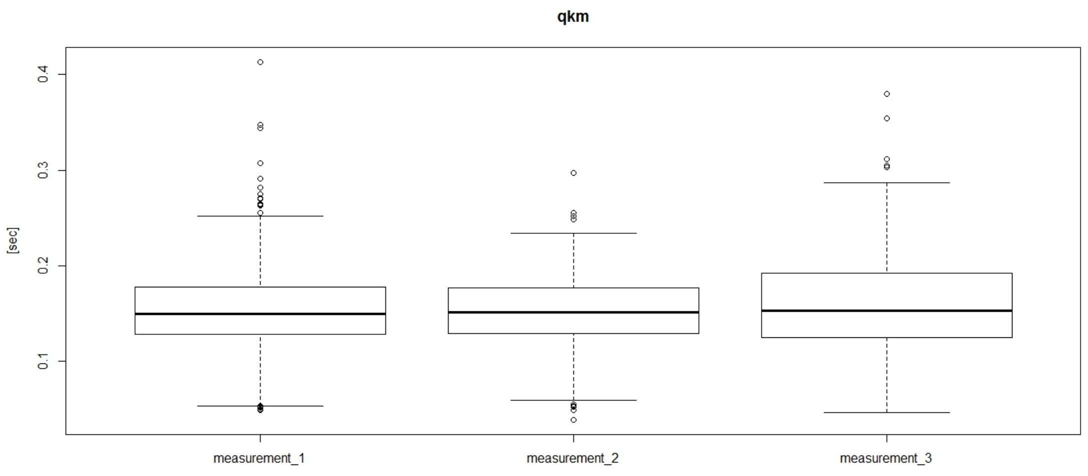


Figure 6.15: Key Generation Time of the Key Store

Key Time [ms]	M_1	M_2	M_3
Lower Whisker	53.46	59.29	46.08
Lower Quartile	127.96	129.42	124.91
Median	149.83	151.51	152.89
Upper Quartile	177.87	176.94	192.41
Upper Whisker	251.78	233.58	287.04

Figure 6.16: Experimental Evaluation of the Key Generation Time of the Key Store

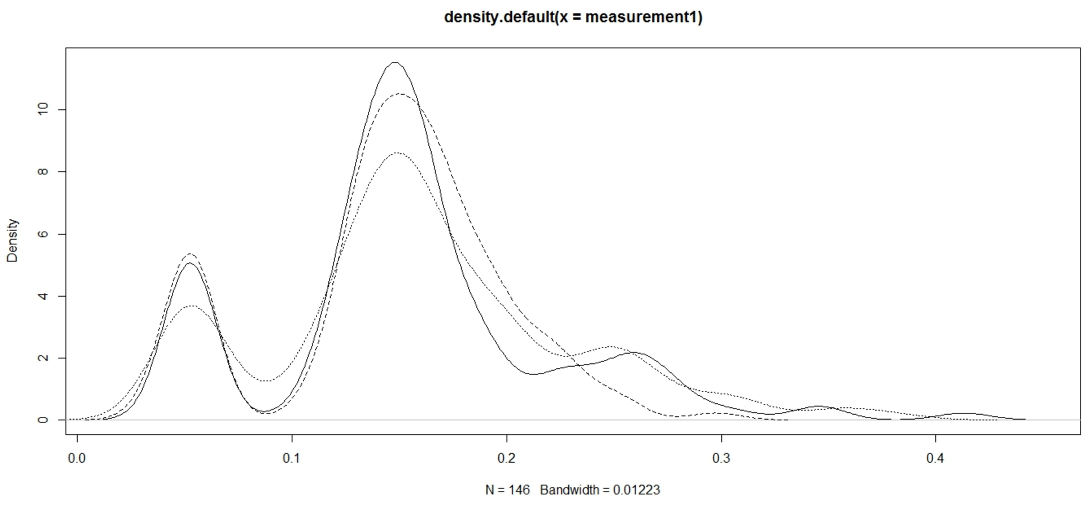


Figure 6.17: Probability Density Function of the Key Generation Time of the Key Store

CHAPTER 7

Conclusion

Network communications will be transmitted by the use of IPv6 instead of IPv4 over wide areas because of the lack of addresses in the domain name system, in the future. There are present classical solutions to ensure information security such as the Diffie Hellman key agreement protocol. The security of a classical cryptographic system is based on the computational complexity of unproven hardness assumptions. Quantum key distribution is an interesting technology to meet future security demands beyond mathematical functions. It provides the secret key generation over the information exchange on a quantum channel via the transmission of qubits between two communication points in a provable secure way and obtains final key material via classical solutions of postprocessing procedures. By contrast to classical solutions, quantum key distribution provides unconditional security. However, it is problematic if there is not sufficient final key material available in the key store to encrypt plaintexts information theoretically secure, today. If the key material is not available the corresponding devices need to delay the communication until the quantum keys have been generated or a classical cryptographic method is chosen. Hence it is necessary to generate preventive key material and not only on demand. The use of quantum key distribution over quantum networks by encrypting plaintexts via one time pads enables unconditional secure cryptographic systems, despite the fact that there exists an unknown quantity of unreliable nodes in the quantum network. Short distance quantum channels implement quantum networks and can be integrated into existing optical fibre infrastructures to enhance its security. The quantum point to point protocol delivers key material over a quantum network based on the BB84 quantum key exchange protocol. We have described the basic mathematics of quantum key distribution and the enhancement of IPv6 by QKD. By implementing a hybrid system in combination with quantum key distribution and classical solutions, we can secure future internet applications information theoretically secure and reduce the weaknesses of quantum channels. We have shown that IPv6 and quantum key distribution complement each other because of the modular design of IPv6 and a quantum network and that quantum key distribution is slower than classical solutions by performance comparisons. However, the quantum extension enables provable secure communication between two distant communication nodes over wide areas but QKD is a subject to interferences from the environment and losses on the quantum channel.

APPENDIX A

Basic Mathematics

Definition A.1 (Cartesian Product of two sets) *The Cartesian product (or product set) of two sets $\mathcal{A}, \mathcal{B}$ is defined by*

$$\mathcal{A} \times \mathcal{B} = \{(a, b) \mid a \in \mathcal{A} \wedge b \in \mathcal{B}\}$$

and is the set of all possible ordered pairs (a, b) which are characterized by

$$(a, b) = (c, d) \Leftrightarrow a = c \wedge b = d$$

If $\mathcal{A}$ and $\mathcal{B}$ are finite, the cardinality of $\mathcal{A} \times \mathcal{B}$ is given by $card\,(\mathcal{A} \times \mathcal{B}) = (card\,\mathcal{A}) \times (card\,\mathcal{B})$.

Definition A.2 (n-ary Relation) *An n-ary relation R over the sets $A_1, \dots A_n$ is a subset of $A_1 \times \dots \times A_n$, i.e. , $R \subseteq A_1 \times \dots \times A_n$.*

For binary relations $(n = 2)$ we also write aRb instead of $(a, b) \in R$ (infix notation).

Definition A.3 (Properties of Binary Relations) *Let $R \subseteq A \times A$ be a binary relation. R is*

reflexive,	*iff* $\forall a \in A\ aRa$
irreflexive,	*iff* $\forall a \in A\ \neg aRa$
symmetric,	*iff* $\forall a, b \in A\ (aRb \Rightarrow bRa)$
antisymmetric,	*iff* $\forall a, b \in A\ (aRb \wedge bRa \Rightarrow a = b)$
transitive,	*iff* $\forall a, b, c \in A\ (aRb \wedge bRc \Rightarrow aRc)$
linear,	*iff* $\forall a, b \in A\ (aRb \vee bRa)$
functional,	*iff* $\forall a, b, c \in A\,(aRb \wedge aRc) \Rightarrow b = c$

Definition A.4 (Function) *Let $\mathcal{A}$ and $\mathcal{B}$ be two non-empty sets. A function (mapping) f from $\mathcal{A}$ to $\mathcal{B}$, written $f\colon \mathcal{A} \to \mathcal{B}$ assigns to each element $a \in \mathcal{A}$ an element $f(a) \in \mathcal{B}$. It can be viewed as a functional relation $R_f \subseteq \mathcal{A} \times \mathcal{B}$ defined by $R_f = \{(a, f\,(a)) \mid a \in \mathcal{A}\}$.*

Definition A.5 (Injection) *A function $f\colon \mathcal{A} \to \mathcal{B}$ is injective or an injection if for all $b \in \mathcal{B}$ there exists at most one $a \in \mathcal{A}$ with $b = f(a)$, i.e. , $a \neq a^{'}$ implies $f(a) \neq f\left(a^{'}\right)$ for all a and $a^{'}$.*

Definition A.6 (Surjection) *A function $f\colon \mathcal{A} \to \mathcal{B}$ is surjective or a surjection if for all $b \in \mathcal{B}$ there exists at least one $a \in \mathcal{A}$ with $b = f(a)$.*

Definition A.7 (Bijection) *A function $f\colon \mathcal{A} \to \mathcal{B}$ is bijective or a bijection if it is injective and surjective, i.e. for all $b \in \mathcal{B}$ there exists exactly one $a \in \mathcal{A}$ with $b = f(a)$.*

Definition A.8 (Inverse) *A function $f^{-1}\colon \mathcal{B} \to \mathcal{A}$ is the reverse (inverse) function of $f\colon \mathcal{A} \to \mathcal{B}$, if $f^{-1} \circ f = id_{\mathcal{A}}$ and $f \circ f^{-1} = id_B$ holds ($id_{\mathcal{A}}$ denotes the identity on a set $\mathcal{A}$, i.e. $id_{\mathcal{A}}(a) = a,$ for all $a_i \in \mathcal{A}$.*

Definition A.9 (Factorial) *The factorial of a non-negative integer n, denoted by $n!$ (n-factorial) is the product of all positive integers less than or equal to n:*

$$0! = 1, \quad (n+1)! = (n+1) \cdot n!$$

Definition A.10 (Permutation) *Let $\mathcal{A}$ be a nonempty finite set. A total bijective function $\pi\colon \mathcal{A} \to \mathcal{A}$ is called a permutation of $\mathcal{A}$.*

A set $\mathcal{A}$ of size n has $n!$ permutations.

Theorem A.1 (Integer Factorization) *Every natural number $n > 1$ can be written as the product of prime numbers.*

$$n = \prod_{k=1}^{m} p_k^{\alpha_k}$$

The numbers p_k are called prime factors and α_k their multiplicities in n. α_k is also denoted as $\nu_{p_k}(n)$.

Definition A.11 (Greatest Common Divisor) *Let $a_1, a_2, \ldots, a_n$ be arbitrary integers s.t. $a_i \neq 0$ for at least one one i. The greatest number in the set of common divisors of $a_1, a_2, \ldots, a_n$ is called the greatest common divisor (gcd) of $a_1, a_2, \ldots, a_n$ and is written as $gcd\,(a_1, a_2, \ldots, a_n)$. Additionally we define $gcd\,(0, \ldots, 0) = 0$. If we know the canonical prime factorization of $a_1, a_2, \ldots, a_n$, the greatest common divisor is given by*

$$gcd\,(a_1, a_2, \ldots, a_n) = \prod_{p\ prime} p^{\underset{i}{min}\{\nu_p(a_i)\}}.$$

Definition A.12 (Coprime Integers) *Integers a, b are called coprime or relatively prime, written as $a \perp b$, if $gcd\,(a, b) = 1$.*

Definition A.13 (Complex Numbers) *The set $\mathbb{C}$ of complex numbers is the set of all numbers that can be written as $z = a + ib$, where $a, b \in R$ and $i^2 = -1$. $a = \mathfrak{Re}\,(z)$ is called the real part, $b = \mathfrak{Im}\,(z)$ the imaginary part of z and i imaginary unit. Furthermore the trigonometric form of a complex number is denoted as follows*

$$z = \rho\,(cos\varphi + i\ sin\varphi)$$

Moreover

$$|z| = \rho\ s.t.\,(0 \leq \rho < \infty)$$

denotes the absolute value of z and

$$arg\ z = \varphi + 2k\pi \quad s.t.\ -\pi < \varphi \leq +\pi, k = 0, \pm 1, \pm 2, \ldots$$

will be referred to as the argument of z.

Definition A.14 (Equality) *Two complex numbers z_1, z_2 are equal iff*

$$\mathfrak{Re}\,(z_1) = \mathfrak{Re}\,(z_2)$$
$$\mathfrak{Im}\,(z_1) = \mathfrak{Im}\,(z_2)$$

holds.

Definition A.15 (Multiplication of Complex Numbers) *The multiplication of complex numbers $z_1, z_2 \in \mathbb{C}$ is defined as*

$$z_1 z_2 = (a_1 + ib_1)\,(a_2 + ib_2) = (a_1 a_2 - b_1 b_2) + i\,(a_1 b_2 + b_1 a_2)\,.$$

Definition A.16 (Division of Complex Numbers) *The division of complex numbers $z_1, z_2 \in \mathbb{C}$ is defined as*

$$\frac{z_1}{z_2} = \frac{a_1 + ib_1}{a_2 + ib_2} = \frac{a_1 a_2 + b_1 b_2}{a_2^2 + b_2^2} + i\frac{a_2 b_1 - a_1 b_2}{a_2^2 + b_2^2}.$$

Definition A.17 (Complex Conjugate) *The complex conjugate of a complex number $z = a+ib$ is defined as*

$$\overline{z} = a - ib$$

Theorem A.2 *The complex numbers with addition and multiplication form a field.*

APPENDIX B

Technical Appendix

B.1 Virtual Media Settings

Machine uuid

```
<Machine uuid="{38153a88-e614-43ec-98d5-3123a6221e12}"
 name="FILENAME" OSType="Linux"
 snapshotFolder="Snapshots"
 lastStateChange="2012-07-25T03:24:50Z">
```

Hard Disk uuid

```
<HardDisks>
 <HardDisk uuid="{8d054f65-3550-4f32-be11-ef5e7cb53afc}"
  location="FILENAME.vdi" format="VDI" type="Normal"/>
</HardDisks>
```

Memory RAM Size

```
<Memory RAMSize="1024" PageFusion="false"/>
```

Network Interface

```
<BridgedInterface name="eth0"/>
```

Cloning a harddisk

```
~\$ VBoxManage clonehd SOURCENAME.vdi
    DESTINATIONNAME.vdi
```

Searching for the uuid

```
~\$ diff ALICE.vbox BOB.vbox | grep "Machine␣uuid"


<   <Machine uuid="{277bb355-4708-4231-9620-fd370c7826cc}"
name="ALICE" OSType="Debian" snapshotFolder="Snapshots"
lastStateChange="2012-08-13T09:03:53Z">
>   <Machine uuid="{d346e246-68d1-464e-af92-1798039391e0}"
name="BOB" OSType="Debian" snapshotFolder="Snapshots"
lastStateChange="2012-08-13T09:04:04Z">
```

B.2 OpenWRT

Target Profile

	Parameter	Description
(X)	Generic	Generic File System

Target Images

	Parameter	Description
(X)	ext4	Root filesystem image
(X)	jffs2	Journalling Flash File System version 2
(X)	squashfs	.sfs filesystem
(X)	Build Grub Images	Boot loader image file
(X)	Use Console Terminal	Console Terminal Support
(X)	Build Virtualbox Image Files (VDI)	Virtual Disk Image file
(X)	Build VMWare Image Files (VMDK)	Virtual Machine Ware Image file
(0)	Seconds to wait before booting the default entry	reduce boot delay
	(/dev/sda2)	root partition on target device

Base System

	Parameter	Description
(X)	base-files	Base filesystem for OpenWrt
(X)	busybox	Core utilities for embedded Linux
(X)	dnsmasq	A lightweight DNS and DHCP server
(X)	drop bear	Small SSH2 client/server
(X)	firewall	OpenWrt firewall
(X)	mtd	Update utility for firmwire updates
(X)	opkg	package manager
(X)	wireless-tools	Tool for manipulating Linux Wireless Extensions

IPv6

	Parameter	Description
(X)	6in4	IPv6-in-IPv4 configuration support
(X)	6scripts	IPv6 scripts
(X)	6to4	IPv6-to-IPv4 configuration support
(X)	6tunnel	IPv4 / IPv6 tunnel proxy
(X)	dhcp6client	IPv6 DHCP client
(X)	dhcp6server	IPv6 DHCP server
(X)	ipv6calc	IPv6 addresses calculation (full)
(X)	radvd	IPv6 Routing Advertisment Daemon
(X)	radvdump	IPv6 Routing Advertisment Dumper

IPv6 Firewall

	Parameter	Description
(*)	ip6tables	IPv6 firewall administration tool

IPv6 Traceroute

	Parameter	Description
(X)	traceroute6	An IPv6-based traceroute implementation

B.3 IP Addressing

IPv4 Address

```
\$ sipcalc -a 192.168.0.1
-[ipv4 : 192.168.0.1] - 0

[Classfull]
Host address            - 192.168.0.1
Host address (decimal)  - 3232235521
Host address (hex)      - C0A80001
Network address         - 192.168.0.0
Network class           - C
Network mask            - 255.255.255.0
Network mask (hex)      - FFFFFF00
Broadcast address       - 192.168.0.255

[CIDR]
Host address            - 192.168.0.1
Host address (decimal)  - 3232235521
Host address (hex)      - C0A80001
Network address         - 192.168.0.1
Network mask            - 255.255.255.255
Network mask (bits)     - 32
Network mask (hex)      - FFFFFFFF
Broadcast address       - 192.168.0.1
Cisco wildcard          - 0.0.0.0
Addresses in network    - 1
Network range           - 192.168.0.1 - 192.168.0.1

[Classfull bitmaps]
Network address         - 11000000.10101000.00000000.00000000
Network mask            - 11111111.11111111.11111111.00000000

[CIDR bitmaps]
Host address            - 11000000.10101000.00000000.00000001
Network address         - 11000000.10101000.00000000.00000001
Network mask            - 11111111.11111111.11111111.11111111
Broadcast address       - 11000000.10101000.00000000.00000001
Cisco wildcard          - 00000000.00000000.00000000.00000000
Network range           - 11000000.10101000.00000000.00000001 -
                          11000000.10101000.00000000.00000001

Network                 - 192.168.0.0     - 192.168.0.0
Network                 - 192.168.0.1     - 192.168.0.1 (current)
...
Network                 - 192.168.0.254   - 192.168.0.254
Network                 - 192.168.0.255   - 192.168.0.255
```

IPv6 Address

```
\$ sipcalc -a -t 2001:0db8:85a3:08d3:1319:8a2e:0370:7347/57
-[ipv6 : 2001:0db8:85a3:08d3:1319:8a2e:0370:7347/57] - 0

[IPV6 INFO]
Expanded Address        - 2001:0db8:85a3:08d3:1319:8a2e:0370:7347
Compressed address      - 2001:db8:85a3:8d3:1319:8a2e:370:7347
Subnet prefix (masked)  - 2001:db8:85a3:880:0:0:0:0/57
Address ID (masked)     - 0:0:0:53:1319:8a2e:370:7347/57
Prefix address          - ffff:ffff:ffff:ff80:0:0:0:0
Prefix length           - 57
Address type            - Aggregatable Global Unicast Addresses
Network range           - 2001:0db8:85a3:0880:0000:0000:0000:0000 -
                          2001:0db8:85a3:08ff:ffff:ffff:ffff:ffff

[V4INV6]
```

```
Expanded v4inv6 address - 2001:0db8:85a3:08d3:1319:8a2e:3.112.115.71
Compr. v4inv6 address   - 2001:db8:85a3:8d3:1319:8a2e:3.112.115.71

[IPV6 DNS]
Reverse DNS (ip6.arpa)  -
7.4.3.7.0.7.3.0.e.2.a.8.9.1.3.1.3.d.8.0.3.a.5.8.8.b.d.0.1.0.0.2.ip6.arpa.

-

\$ sipcalc -a 0:0:0:0:0:0:0:0
-[ipv6 : 0:0:0:0:0:0:0:0] - 0

[IPV6 INFO]
Expanded Address        - 0000:0000:0000:0000:0000:0000:0000:0000
Compressed address      - ::
Subnet prefix (masked)  - 0:0:0:0:0:0:0:0/128
Address ID (masked)     - 0:0:0:0:0:0:0:0/128
Prefix address          - ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Prefix length           - 128

\$ sipcalc -a 0000:0000:0000:0000:AC43:9876:1234:5555
-[ipv6 : 0000:0000:0000:0000:AC43:9876:1234:5555] - 0

[IPV6 INFO]
Expanded Address        - 0000:0000:0000:0000:ac43:9876:1234:5555
Compressed address      - ::ac43:9876:1234:5555
Subnet prefix (masked)  - 0:0:0:0:ac43:9876:1234:5555/128
Address ID (masked)     - 0:0:0:0:0:0:0:0/128
Prefix address          - ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Prefix length           - 128

\$ sipcalc -a 3228:00A3:0000:0000:0221:01FF:CE01:7842
-[ipv6 : 3228:00A3:0000:0000:0221:01FF:CE01:7842] - 0

[IPV6 INFO]
Expanded Address        - 3228:00a3:0000:0000:0221:01ff:ce01:7842
Compressed address      - 3228:a3::221:1ff:ce01:7842
Subnet prefix (masked)  - 3228:a3:0:0:221:1ff:ce01:7842/128
Address ID (masked)     - 0:0:0:0:0:0:0:0/128
Prefix address          - ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

\$ sipcalc -a 4AC0:0:0:7:0:0:0:C
-[ipv6 : 4AC0:0:0:7:0:0:0:C] - 0
```

```
[IPV6 INFO]
Expanded Address        - 4ac0:0000:0000:0007:0000:0000:0000:000c
Compressed address      - 4ac0:0:0:7::c
Subnet prefix (masked)  - 4ac0:0:0:7:0:0:0:c/128
Address ID (masked)     - 0:0:0:0:0:0:0:0/128
Prefix address          - ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Prefix length           - 128
```

Bibliography

[1] Maarten van Steen Andrew S. Tanenbaum. *Verteilte Systeme Prinzipien und Paradigmen*. Pearson Studium, 2 edition, 2008.

[2] C. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin. Experimental quantum cryptography. *Journal of Cryptology*, 5:3–28, 1992.

[3] C. Bennett and G. Brassard. Quantum cryptography: public key distribution and coin tossing. pages 175–179, Bangalore, 1984. IEEE.

[4] Gilles Brassard, Isaac Chuang, Seth Lloyd, and Christopher Monroe. Quantum computing.

[5] Bronstein, Semendjajew, Musiol, and Mühlig. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, 5 edition, 2001.

[6] Internet Systems Consortium. Internet domain survey. `http://ftp.isc.org/www/survey/reports/current/`, 2012. [Last access 2012-09-24].

[7] Internet Systems Consortium. Internet domain survey - host count history. `https://www.isc.org/solutions/survey/history/`, 2012. [Last access 2012-09-24].

[8] G. Van de Velde, C. Popoviciu, T. Chown, O. Bonness, and C. Hahn. Ipv6 unicast address assignment considerations. RFC 5375, December 2008.

[9] S. Deering and R. Hinden. Internet protocol, version 6 (ipv6) specification. RFC 2460 (Draft Standard), `http://www.ietf.org/rfc/rfc2460.txt`, December 1998. Updated by RFCs 5095, 5722, 5871.

[10] S. Deering and R. Hinden. Ip version 6 addressing architecture. RFC 4291 (Draft Standard), `http://www.ietf.org/rfc/rfc4291.txt`, February 2006.

[11] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. 400:97–117, 1985.

[12] David Deutsch and Richard Jozsa. Rapid solutions of problems by quantum computation. 439(1907):553–558, 1992.

[13] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.

[14] Gittenberger B. Karigl G. Panholzer A. Drmota, M. *Mathematik für Informatiker*. Heldermann Verlag, 2007.

[15] Robert G. Gallager. Low-density parity-check codes, 1963.

[16] N. Gisin, G. G. Ribordy, W. Tittel, and H. Zbinden. Quantum cryptography. *Reviews of Modern Physics*, 74:145–195, 2002.

[17] W. Greiner. *Quantenmechanik Einführung*. Verlag Harri Deutsch, 2005.

[18] Michael Gutmann and Detlef Lannert. *Linux im Netzwerk*. Addison-Wesley Verlag, 2007.

[19] T. Hardie. Distributing authoritative name servers via shared unicast addresses. RFC 3258 (Informational), April 2002.

[20] D. Harkins and D. Carrel. The internet key exchange (ike). RFC 2409 (Standards Track), November 1998.

[21] D. Haskin and E. Allen. Ip version 6 over ppp. RFC 2472 (Standards Track), December 1998.

[22] Red Hat. Main page - kvm. `http://www.linux-kvm.org/`, 2012.

[23] R. Hinden, S. Deering, and E. Nordmark. Ipv6 global unicast address format. RFC 3587 (Informational), August 2003.

[24] R. Hinden and B. Haberman. Unique local ipv6 unicast addresses. RFC 4193 (Proposed Standard), October 2005.

[25] Eduard Hirsch. Quantum cryptography applied for safety relevant systems. Master's thesis, Vienna University of Technology, 2008.

[26] The Internet Engineering Task Force (IETF). `http://www.ietf.org/`, 2012. [Last access 2012-09-24].

[27] J.M. Jauch and F. Rohrlich. *The Theory of Photons and Electrons*. Springer-Verlag, 1980.

[28] C. Kaufman. Internet key exchange (ikev2) protocol. RFC 4306 (Proposed Standard), December 2005.

[29] S. Kent. Ip encapsulating security payload (esp). RFC 4303, December 2005.

[30] S. Kent and R. Atkinson. Ip authentication header. RFC 2402 (Standards Track), November 1998.

[31] S. Kent and R. Atkinson. Security architecture for the internet protocol. RFC 2401 (Standards Track), November 1998.

[32] Christian Kollmitzer and Mario Pivk. *Applied Quantum Cryptography*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[33] H.-J. Kowalsky. *Lineare Algebra*. Walter de Gruyter Berlin, 9. edition, 1979.

[34] A. Liakopoulos, D. Kalogeras, V. Maglaris, D. Primpas, and C. Bouras. Qos experiences in native ipv6 networks. *International Journal of Network Management*, 19(2):119–137, 2009.

[35] N. David Mermin. *Quantum Computer Science - An Introduction*. Camb, 2007.

[36] Microsoft. Microsoft virtualization. `http://www.microsoft.com/`, 2012.

[37] Gernot Münster. *Quantentheorie*. Walter de Gruyter Berlin, 2006.

[38] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 3rd edition, 2002.

[39] OpenWRT. Openwrt - wireless freedom. `https://openwrt.org/`, 2012. [Last access 2012-08-28].

[40] ORACLE. Oracle vm virtualbox. `https://www.virtualbox.org/`, 2012. [Last access: 2012-08-28].

[41] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[42] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

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

[44] Peter W. Shor and John Preskill. Simple proof of security of the bb84 quantum key distribution protocol. *Phys. Rev. Lett.*, 85:441–444, Jul 2000.

[45] J. Smith and R. Nair. *Virtual Machines: Versatile Platforms for Systems and Processes (The Morgan Kaufmann Series in Computer Architecture and Design)*. Denise E. M. Penrose, 2005.

[46] William Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall Press, 5th edition, 2010.

[47] William Stallings. *Operating Systems - Internals and Design Principles*. Pearson Prentice Hall, 7 edition, 2011.

[48] VMWare. Vmware virtualization software for desktops, servers, virtual machines for public and private cloud solutions. `http://www.vmware.com/`, 2012. [Last access: 2012-08-28].

[49] Andrew Whitaker. *Einstein, Bohr and the Quantum Dilemma*. Cambridge University Press, New York, 2 edition, 2006.

[50] H. Xu, L. Ma, A. Mink, B. Hershman, and X. Tang. 1310-nm quantum key distribution system with up-conversion pump wavelength at 1550nm. *Optics Express*, 15:7247–7260, June 2007.