Quantum Networking: Explore QKD and quantum internet

Project for

Advanced Networks Master's program course

Written By: Nikolaos Mouzakitis MTP321

Date Last Edited: January 22, 2025

Abstract

Quantum networking represents a transformative advancement in secure communication, with Quantum Key Distribution (QKD) and the quantum internet standing at its core. Unlike the widely available and adopted classical systems, QKD leverages certain mechanical principles from the field of quantum physics in order to theoretically guarantee unbreakable encryption schemes, addressing critical vulnerabilities in modern cryptographic methods. In the early days of its rise, quantum internet promises a global network enabling unprecedented levels of secure communication, distributed quantum computing, and advanced scientific applications.

Contents

Chapter 1

Introduction

Quantum networking stands as the next frontier in communication technology, which is expected to leverage principles of quantum mechanics for achieving unprecedented levels in security, as well as in computational capability. In contrast to ordinary classical networks, quantum networks rely on quantum states of the likes of superposition and entanglement to achieve operations impossible for the classic networking schemes. These attributes become of vital importance in application areas like secure communications, where quantum key distribution (QKD) promises provable security against eavesdropping, and in distributed computing in cases where quantum nodes could collaborate to solve problems beyond the reach of classical networks. In the modern world, where the requirement for global data security keeps following an increasing trend, quantum networking is considered as a robust solution for protecting sensitive informations. Moreover, it is possible that mixed architectures combining classical and quantum network elements could provide a feasible solution for the future communication systems. To define the structure and complete functionality of the future quantum internet, a complete network stack is required to be created starting from the beginning utilizing the features of quantum entanglement[12].

Challenges arise although; despite the promising potential in quantum networking, it faces technical and theoretical obstacles that need to be surpassed in order to achieve and enable an entirely practical implementation and adoption. Because of the very nature of quantum mechanichs, quantum internet is opted to utilize concepts without classical counterparts with the likes of quantum entanglment, no-cloning theorem, quantum measurement and teleportation.

In contrast to the classical and well known model of computation experienced so far, where we deeply rely on the fact that information can be read and copied, this concept that does not hold for quantum networking.

The scalability remains one of the main issues, as current quantum networks are limited in the number of nodes and the distances they can span without degradation. This limitation comes from the fragile nature of the quantum states, which are very sensitive on environmental noise and decoherence especially over long distances. As a countermeassure, in order to avoid these limitations the development of reliable quantum repeaters and advanced error-correction schemes are required. Additionally, the efficient generation, distribution and storage of entanglement across a network has its difficulties and limitations that are required to be surpassed, as maintaining a high fidelity value in entangled states is necessary for a complete and functioning network. In the end, another challenge lies in integration quantum systems with the classic infrastructure, which requires coordination between two different operational paradigms. Addressing these challenges is critical for the transission of quantum networking from the experimental level to real-world applications.

In this project the current state of quantum networking is explored alongside its potential applications. The introduction provides a broad overview of the field, emphasizing its importance in secure communication and presents a brief description of the most common terminology of quantum networking. In the related work section recent advancements in quantum networking are presented, with a focusing on technologies such as quantum key distribution, entanglement distribution, and quantum repeaters. In the combined methodology and results section, a simulation-based approach utilizing QuTiP showcases examples to study the key elements of quantum networking, presenting the findings from the simulations. The discussion section interprets these findings in the context of existing literature, identifying both strengths and limitations of current technologies. In the end, the conclusion section summarizes the project's work.

1.1 A brief description of the most common terms in quantum networking

A **quantum network** consists of a set of quantum processors/devices connected to each other. **Qubits** can be particles such as photons or electrons, carrying certain properties derived from quantum mechanics called *state*. A state's can express the spin of an electron or a photon's polarization. The particular state of a qubit is unknown prior to its measurement, since it is in a superposition (*as an analogy one can think the linear combination of two variables)* of states. After its measurement the qubit collapses in one state, which actually can not be determined before since it is related with the basis that is used in the measurement.

Quantum superposition consists one of the ground principles in quantum mechanics. It presents the property that quantum system has, of existing in multiple states at the same time, until it gets measured. Superposition denotes that prior to measurement, the system does not exist in any state but in a combination of both states. States in quantum mechanics are denoted as $|\psi\rangle$ and represent a vector in a complex Hilbert space. A system in superposition is described as a linear combination of basis states:

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle,$$

where:

- $|0\rangle$ and $|1\rangle$ are the basis states, and
- a_0 ,a_1 belong to complex numbers and satisfy the following equation:

$$|a_0|^2 + |a_1|^2 = 1.$$

Coefficient $|a_0|^2$ represent the probability of measuring the system state as $|0\rangle$ and $|a_1|^2$ the probability to measure it as $|1\rangle$. After describing its properties, we can understand why superposition constitutes a basic difference in between quantum and classical mechanics.

Quantum entanglement is the situation where two qubits become correlated (called *entanglement pair*) in a way that any modification on the state of the first at once affect the state, and is reflected on the second qubit, without any consideration of their actual distance. Because of the no-cloning theorem, there it is not even possible for a third qubit to be entangled with either of those qubits. Close tied to entanglement, the concept of **quantum teleportation** is a process, in which state information of a quantum particle can be transferred from one location into another without that particle beeing trasported. This incident is feasible due to the entanglement.

Another important property is the **fidelity** of a quantum state. Fidelity is a metric $f \in [0, 1]$, denoting the quality of a quantum state. The higher the value is, the closer the quantum state is to what it was created to be. Fidelity represents this probability, allowing the evaluation of how much environmental noise and losses have impacted the quantum state.

The **no-cloning theorem** is a fundamental principle in quantum mechanics which states that it is impossible to create an exact copy of an unknown quantum state. This theorem has important consequences in quantum information theory and especially in quantum cryptography and networking.

The theorem therefore guarantees the secure quantum communication in protocols such as Quantum Key Distribution (QKD) because it is impossible for any eavesdropper who copies quantum states in order to intercept informations to operate unnoticed.

1.2 QKD

Quantum Key Distribution (QKD) is a revolutionary method for securely exchanging cryptographic keys between two distant parties[13], commonly referred to as Alice (the sender) and Bob (the receiver). It utilizes the principles of quantum mechanics domain in order to ensure the security of the key distribution operation. QKD's main goal is to enable the two parties to establish a shared, secret cryptographic key, which can then be used to encrypt and decrypt messages using traditional cryptographical schemes, such as symmetric encryption algorithms.

One of the main differenes that differentiates QKD from other cryptographic schemes, is its reliance on two communication channels: a quantum channel and a classical channel. The quantum channel is used to transmit quantum bits or qubits, which are the fundamental units of quantum information. These qubits are usually encoded as quantum states of particles, for example as photons. The quantum channel ensures the secure transmission of the qubits because, according to the no-cloning theorem and Heisenberg's principle of uncertainty, any interception or measurement of these quantum states will unreversibly disturb the system, revealing the presence of a malicious actor(adversary).

On the other hand, the classical channel is used for the transmission of classic information that does not relates to quantum information but plays vital role for the key establishment process. When the qubits are exchanged via the quantum channel, Alice and Bob communicate using the classical channel to perform the next steps, such as error correction, privacy amplification, and sifting. These processes guarantees that noise introduced during the transmission is considered and that they both agree on a shared key. The classical channel also helps into validating the integrity of the key exchange and for other parameters, such as the measurement bases in some QKD protocols.

The core components on which QKD is based, are the concepts of quantum entanglement and superposition, which makes possible the secure transfer of information. For example, in protocols like BB84, the cryptographic key is distributed using quantum states that are in superposition. The critical feature of QKD's protocols is that any attempt of a malicious actor for eavesdropping will modify inreversible the quantum states, an action which is detectable by Alice and Bob sides. This is the main difference of QKD compared with classical cryptographic schemes, which focus on mathematical complexity.

QKD protocols like BB84 and E91 also offer *information-theoretic security.* This term has the meaning that even if an adversary has unlimited computational resources available, they cannot gain any knowledge about the shared key without being detected. This statement highlights the difference with traditional cryptographic methods, and someone could argue that QKD its secure by the foundations of the "ingredients" its made of.

As for the challenges that QKD has to surpass, quantum channels typically have limitations in terms of distance and reliability. As a solution, quantum repeaters and quantum relays have been introduced for extending the range of QKD. These devices would allow entanglement to be distributed over longer distances, enabling the construction of a desired quantum internet where secure communication can take place over large scale networks.

In general, QKD is an core building block of quantum cryptography, providing a secure way for establishing shared secret keys. By using the principles of quantum mechanics, QKD enables secure communication that is information-theoretic secure. While it still faces some practical challenges, QKD holds high expectations for the future of secure communications.

1.3 Quantum Repeaters

Quantum repeaters are devices which are designed to extend the range of quantum communication networks. They are required to enable QKD in long distances and for supporting the construction of a future quantum internet. In the classical way of communication, the repeaters are used in order to amplify signals for avoiding signal loss, but due to the no-cloning theorem which prohibits duplication of quantum states, this is not applicable on quantum communication. So a quantum repeater in constrast, utilizes entanglement distribution and error correction schemes to achieve the establishment of a quantum connection over long distance. The main issues that quantum repeaters address are the following:

a) Photon Loss (Attenuation): In fact, quantum repeaters can help to tackle the loss in fiber-based quantum communication, where the photons who carry quantum information experience loss inside the medium by propagation. This limits the range of QKD to about tens of kilometers [4]. By using them, and creating shorter segments with the interference of quantum repeaters in the medium, entanglement pairs can be created and saved for each segment, and by using the technique of entanglement swapping, information can 'pass' through and realize a long distance connection.

b) Decoherence: Quantum states are very sensitive and affected by environmental noise, forcing the states to lose coherence (*preservation of quantum state over time and long distances*). Quantum repeaters use quantum memories[14] as intermediate nodes to store entangled states temporarily, allowing processes such as error correction or entanglement purification to maintain the quality of the entanglement.

c) Scalability of Quantum Networks: Without using repeaters, quantum networks have to be limited small local areas due to the constraints of photon transmission. Quantum repeaters can enable a scalable construction of larger quantum networks by bridging long distances using segmented paths with entangled pairs connecting multiple nodes.

d) Error Accumulation: In quantum communications, errors tend to accumulate over long distances and the classical error correction schemes are not applicable for the quantum states. Quantum repeaters integrate error correction protocols like entanglement purification in order to tackle this issue.

1.4 Quantum Switches

A quantum switch is a fundamental device of the quantum networking architecture which enables two operations(events) to be performed in a *quantum superposition of different orders*. It has the means to dictate the ordering of quantum processes and uses the rule of quantum superposition to make the sequence of events not decided prior to an observation or measurement.

In classical systems, the order of operations is fixed: we can know with certainity that one event follows another in sequence. In contrast, with quantum switches, the order of the execution of events is also in a superposition. As an example, Let A and B, two events-actions to be performed. Quantum switch can create a state where both "perform A first, then B" and "perform B first, then A" exist in superposition.

This kind of superposition(of different orders) makes occuring event's order to be not well defined until a measurement occurs. This feature enhances communication capabilities in quantum systems as authors in [23] shown that utilizing quantum superposition of different orders in 2 noisy channels, can enable transmittion on these mediums achieving high quality results, even if individually no information could be transfered in any of these channels if used standalone.

Summarizing, the quantum switches are a novel tool in quantum networking, which makes use of the superposition of order to generate new possibilities for quantum communication in ways unachievable by design for classical communications.

1.5 Bell state measurement

A Bell State Measurement (BSM) is a quantum operation designed to determine the entangled state of two qubits. The set of Bell states consists of four maximally entangled quantum states representing the strongest correlations among two qubits. These states are:

$$\begin{aligned}
\Phi^+ &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \\
\Phi^- &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \\
\Psi^+ &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \\
\Psi^- &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).
\end{aligned}$$

In a BSM operation, the states of two qubits are projected onto one of these four Bell states. As an outcome, information occurs about how the two qubits are related, but not information about their respective individual states. Their ability to manage and process entanglement in quantum networks establishes BSMs to be a critical element on the future quantum architectures.

1.6 Proposed Quantum OSI-like Models

Quantum Internet(QI) architecture research has been studied and lead into several OSI-like protocol stack models, where the common aspect of all has been the exploitation of quantum entanglement. Entanglement have been considered to be the key part for communication. These models, such as proposed by Van Meter et al.[2], Wehner et al.[10], Li et al. [9], and Dür et al.[20], consider the entanglement generation and management-distribution through various methods. One basic categorization of these models is that Dür's model utilizes a multipartite entanglement fashion while the other models follow a bipartite entanglement approach.

In Van Meter et al.'s model, authors describe a layered protocol for quantum repeaters network and they implement end to end entanglement distribution and different 'actions' of entanglement per layer. They underlined bipartite entanglement generation using protocols and error purification techniques in between the layers to achieve high fidelity in order to support the quantum communication. Similarly, in Wehner et al.'s model, the classical Internet layer names are present with the introduction of abstractions that can support hardware heterogeneity and scalable entanglement generation. Li et al.'s approach extends this, by pre-establishing entangled states that can be consumed on demand, while customizing link-layer protocols for maintaining high-fidelity connections. In contrast, the model presented by Dür et al., introduces multipartite entanglement and graph states, which can support long range entanglement distribution by utilizing dynamic and adaptive network steps. As we can inspect there is a diversity of ways for the proposed models, having single-pair entanglement, multi-party sharing of entanglement and advanced graph structures, competing for the model to be used in the quantum Internet network stack.

As for the main network entities of the quantum networking paradigm[7] are presented in Table 1.1.

Entity	**Role**
quantum end nodes	able to transmit and process qubits for quantum applications
quantum memories	enabling reliable storage of qubits for later operation or measurement
quantum routers	routing of entanglement distributions, responsible to choose optimal swapping paths
quantum switches	able to achieve transfer of high quality information, surpassing obstacles such as noise by utilizing quantum superposition of ordering
quantum repeaters	extending network by purifying, entanglement swapping and delivering high fidelity entanglement
EPR*Einstein,Podolsky,Rosen* Sources	Generate and manage distribution of entangled qubit pairs among neighbouring nodes for creation of entanglement links. They emit engtangled particles and serve as backbone for transfering quantum data across nodes.
Physical medium	Fiber optics or free space channels to enable qubits transmission

Table 1.1: Network entities of the quantum networking paradigm

Chapter 2

Related Work

In recent years, quantum networking has gained significant attention due to its potential to ignite a revolution in secure communications and distributed computing. Khristo et al. [1] have provided an overview of the field's current status but also of the desired future directions. In their work the key advancements and challenges in the development of quantum networks have been discussed while in addition explored the integration of the quantum key distribution (QKD) and presented a broader vision for the quantum internet. Valls and co-authors [3], discuss quantum operations for network control and present a model for multi-hop distribution of entanglement among the nodes of a quantum system, functionalities that need to be efficient for the support of distributed quantum systems.

Van Meter et al. [2] proposed a complete architecture for a scalable unified quantum communications. They utilize a two-pass connection setup and recursive protocol to allow scalability and extensibility for future internetworking of many quantum devices. Authors also presented protocols-algorithms for connection establishments, desicion making, routing and allocation of resources inside the quantum network and its consisting layers. In another step towards the realization of a complete quantum network architecture, researchers have implemented complete protocol stacks for quantum networks. Pompili et al. [6] have experimented with entanglement delivery across a network's protocol stack and proposed a division of it, in order to provide independency between the physical implementation and the uppermost layers. In this way the protocol stack is splitted into Physical and Platform-Independent (Hardware abstraction layer, Link layer, Network layer, Application layer). A thouroughly presentation of the quantum network architecture is given by Khan et al. [7] where they have studied the quantum protocol stack, and underlined the functionalities of its layers, as well as its components. In their work authors proposed E-QNET, a prototype for created to be utilizing a layered quantum networking framework.

In[9], Li and co-authors proposed a new way of how interconnected quantum nodes can communicate by adopting an OSI-like model concept for QI. Their model intends to reduce the complexity of communications and proposed the QLAN (quantum local area network) as a basic component of the future QI. In their model QLANs are interconnected via intermediate quantum repeaters, and they can also be designed and deployed in a hierarchical fashion. Kozlowski et al. [12] focus on presenting the architectural foundations required in order to realize the quantum internet, strongly proposing the incorporation of quantum entanglement and Bell pairs for quantum networks. The different aspects among quantum and classical networks are discussed, while authors point out the use of hybrid classical-quantum architectures. Quantum internet concept is also been explored by Cacciapuoti et al. [5]. In their work the key differences among quantum and classical networks ar underlined, and point out quantum teleportation as the most fundamental method of transporting informations in a quantum system. Authors proposed a framework for tackling the challenges of a quantum network creation as well as emphasizing on the need to address problems like decoherence for enchancing the reliability these networks. The key role of quantum teleportation is also discussed in [11], where authors recommend a redesignation of the classic communication paradigm in order to conform with the quantum teleportation needs. They describe the generation and distribution of entanglement by using photons for transmissions, and propose certain schemes for extending the range of communications such as the entanglement swapping.

Alshowkan et al.[17] focused on the improvement of the infrastructure and performance of quantum local area networks (QLANs) for supporting advanced quantum protocols and communication. They have utilized White Rabbit switches(classical high-precision Ethernet-based switches) for synchronization of remote network nodes with very low timing jitter (picosecond scale). By that, the fidelity of entanglement distribution was enhanced in comparisson to systems using GPS-based synchronization. Authors also integrated a QKD layer for securing classic communication required for control and data management and created a network using already available commercial components, demonstrating the feasibility of their solution for the existing fiber optic infrastructure.

In contrast to traditional quantum repeaters who require the use of entaglement swapping and quantum memory, all-photonic quantum repeaters are based on light particles(photons) for the transfer of quantum information. Authors in [4] propose an all photonic quantum repeater based on RGS (repeater graph states) achieving a higher repetition rate compared to traditional quantum repeaters, and does not require the use of quantum memories. Their proposed scheme offers a fast Bell pair generation ration with only limitation the time required for the creation of the repeater graph state. As per quantum memories, in their work Bhaskar et al. [14], demonstrated a quantum memory node, which was integrated into a nanophotonic resonator. Their memory node, was able to perform asynchronous Bell-state measurements and proved that their setup is outperforming direct methods of transmission. Alongside quantum repeaters, quantum switches also are going to play a vital role in the future quantum networking architecture. Caleffi and Cacciapuoti [16] have theoretically analysed the overhead on entanglement distribution that a quantum networking system without the use of quantum switch experiences in comparisson with the use of quantum switch. By using closed-form equations desribing the entanglement distribution were able to quantify and measure the gain for the average fidelity of the entanglments using quantum switches.

Many fields have the potential to be benefited by a complete quantum networking architecture, such the field of ultra scalable analytics. Quantum computing utilization can transform completely the field and improve efficiency, security and offer more scalable power system analysis. In their reseach, Zhou et. al [13], review the current state of power grid analytics and discuss quantum networking methods that can enable secure communications for power grids in the context of quantum-engineered smart grids. Lv and co-authors[8], consider a conjunction of digital twins and quantum netwroking for enchancing the communication security on Industrial Internet of Things(IIoT). In their work, they demonstrate how QKD can be applied for secure communication in the field of IIoT and explore the integration of quantum networking and digital twins in order to enlight how can the virtual models interact through quantum communication channels.

Chapter 3

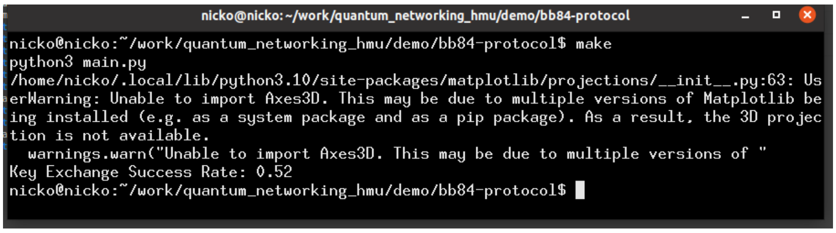
Methodology and Results

For the purposes of this project, a simulation approach using software tools have been employed in order to enable the demonstration of selected concepts in quantum networking. QuTiP(Quantum Toolbox in Python)[21], an open source framework written in Python, has been used for the simulations. Another platform which allows public access is IBM Quantum Platform[22] which is available for cloud-based quantum computing services. Users can have access to a subset of IBM's quantum processors to deploy quantum circuits and other useful resources such as tutorials specified for quantum computing. Quantum processor are also accesible via the IBM Quantum Composer (graphical drag and drop quantum programming tool) where quantum operations for generating quantum circuits are available.

3.1 BB84 Quantum Key Distribution Protocol

BB84 protocol is a foundational work in the field of quantum key distribution (QKD) protocols, been proposed back in 1984 [18]. It realizes a secure communication of a secret cryptographic key alongside two entities, over a potenitally insecure medium/channel by using principles of quantum mechanics.

In this simulation (code is in Listing 1 of Appendix A), two entities entitled Alice and Bob, execute the QDK using the BB84 protocol in order to share a secret key over a potentially non-secure medium. Initially Alice creates a number of qubits (100 in this example) randomly in one of the two possible bases (either 'x' or 'z') and sends them to Bob. Bob on his side, randomly selects a base per qubit and measures it. In the next step, they are comparing their used bases over a classical communication channel and they discard all the bits generated as an outcome from the measured qubits whose bases were not matching. The remaining bits which obtained from matching bases among the two entities are kept and create the shared secret key. It is important to note, that the qubits measurements from the mismatched bases are discarded, since the outcome of the measurement on a mismatched base is probabilistic, and this does not ensure an identical secret key for both parties.


nicko@nicko: ~/work/quantum_networking_hmu/demo/bb84-protocol

```
nicko@nicko:~/work/quantum_networking_hmu/demo/bb84-protocol$ make
python3 main.py
/home/nicko/.local/lib/python3.10/site-packages/matplotlib/projections/__init__.py:63: Us
erWarning: Unable to import Axes3D. This may be due to multiple versions of Matplotlib be
ing installed (e.g. as a system package and as a pip package). As a result, the 3D projec
tion is not available.
  warnings.warn("Unable to import Axes3D. This may be due to multiple versions of "
Key Exchange Success Rate: 0.52
nicko@nicko:~/work/quantum_networking_hmu/demo/bb84-protocol$ 
```

Figure 3.1: Execution of code of Listing 1 of Appendix A.

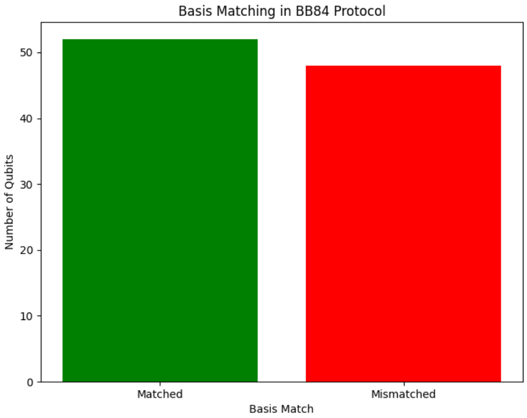


Figure 3.2: Bases matched on the example where 100 qubits used.

As we can observe in Figure 3.3, the relation of success rate is plotted against the utilized number of qubits in the protocol. The simulation was run for up to 10000 qubits. Success rate is converging to value 0.5, as the probability of Alice and Bob choosing the same base (’x’ or ’z’) is 50%.

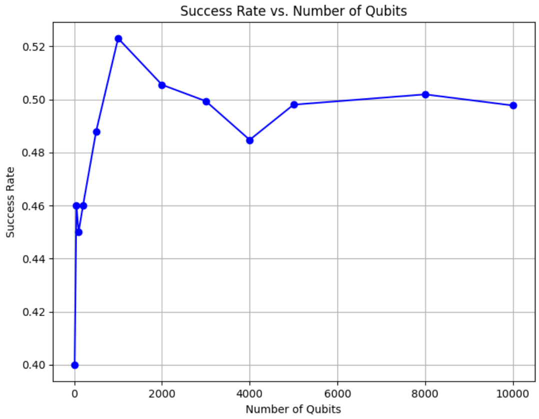


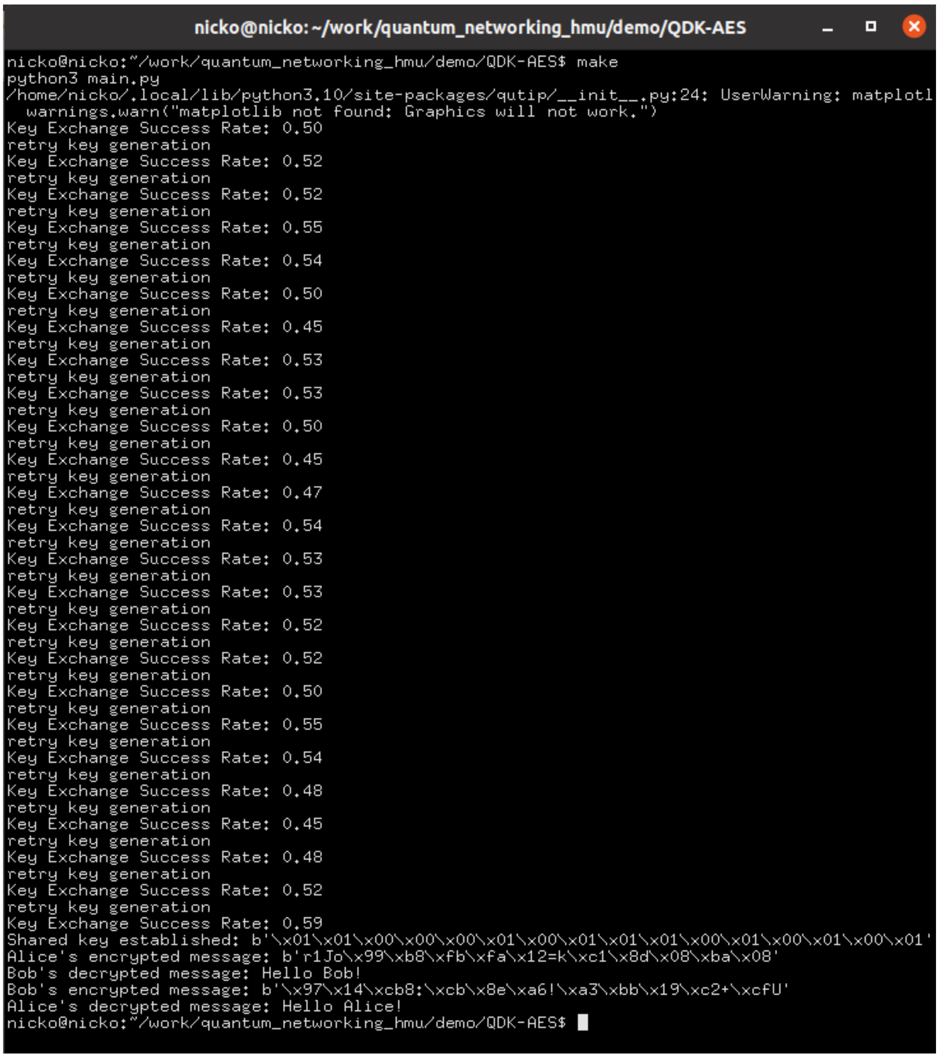
Figure 3.3: Success rate vs number of qubits used.

3.2 QKD application using AES for classic communication

In the Listing 3 of Appendix A, building on the previous example of generating a key, we extend the functionality in encrypted message exchanges among two entities, using the secret key created by the QKD for a symmetric encryption algorithm (in this case AES). Also a limit on the success rate of the key generation is implemented, so each time that the success rate is not satisfied generation restarts. In this example 256 qubits were selected with success rate to be above 0.58. The execution of the simulation is presented on Figure 3.4, one message is sent by each entity simulating an encrypted ping-pong among them. In a real application we could have implemented other features as well such as a key lifetime(how often the AES key should be refreshed) authentication between the entities start the QKD process and error correction as it would be required.

3.3 Fidelity against Noise Level

In this simulation (code is in Listing 2 of Appendix A), a model of a quantum repeater network with entanglement swapping is created in order to observe how the fidelity of the final entangled state is affected by depolarizing noise [19]. Depolarizing noise is a quantum noise which models the coherence loss in quantum system's states because of imperfections in quantum operations.



```
nicko@nicko:~/work/quantum_networking_hmu/demo/QDK-AES$ make
python3 main.py
/home/nicko/.local/lib/python3.10/site-packages/qutip/__init__.py:24: UserWarning: matplotl
  warnings.warn("matplotlib not found: Graphics will not work.")
Key Exchange Success Rate: 0.50
retry key generation
Key Exchange Success Rate: 0.52
retry key generation
Key Exchange Success Rate: 0.52
retry key generation
Key Exchange Success Rate: 0.55
retry key generation
Key Exchange Success Rate: 0.54
retry key generation
Key Exchange Success Rate: 0.50
retry key generation
Key Exchange Success Rate: 0.45
retry key generation
Key Exchange Success Rate: 0.53
retry key generation
Key Exchange Success Rate: 0.53
retry key generation
Key Exchange Success Rate: 0.50
retry key generation
Key Exchange Success Rate: 0.45
retry key generation
Key Exchange Success Rate: 0.47
retry key generation
Key Exchange Success Rate: 0.54
retry key generation
Key Exchange Success Rate: 0.53
retry key generation
Key Exchange Success Rate: 0.53
retry key generation
Key Exchange Success Rate: 0.52
retry key generation
Key Exchange Success Rate: 0.52
retry key generation
Key Exchange Success Rate: 0.50
retry key generation
Key Exchange Success Rate: 0.55
retry key generation
Key Exchange Success Rate: 0.54
retry key generation
Key Exchange Success Rate: 0.48
retry key generation
Key Exchange Success Rate: 0.45
retry key generation
Key Exchange Success Rate: 0.48
retry key generation
Key Exchange Success Rate: 0.52
retry key generation
Key Exchange Success Rate: 0.59
Shared key established: b'\x01\x01\x00\x00\x00\x01\x00\x01\x01\x01\x00\x01\x00\x01\x00\x01'
Alice's encrypted message: b'r1Jo\x99\xb8\xfb\xfa\x12=k\xc1\x8d\x08\xba\x08'
Bob's decrypted message: Hello Bob!
Bob's encrypted message: b'\x97\x14\xcb8:\xcb\x8e\xa6!\xa3\xbb\x19\xc2+\xcfU'
Alice's decrypted message: Hello Alice!
nicko@nicko:~/work/quantum_networking_hmu/demo/QDK-AES$ █
```

Figure 3.4: Simple encrypted message exchange among the two nodes using the key generated by QKD for AES.

The depolarizing noise on a quantum state can be expressed as [15]:

$$\rho \to (1-p)\rho + \frac{p}{d}I$$

where:

- ρ: density matrix of quantum state before the noise is applied.
- p: depolarizing probability, value $\in [0,1]$.
- d: Hilbert's space dimension. For a single qubit, $d = 2$.

- I: identity matrix $d \times d$.

This form to model the noise is used in the presented simulation code.

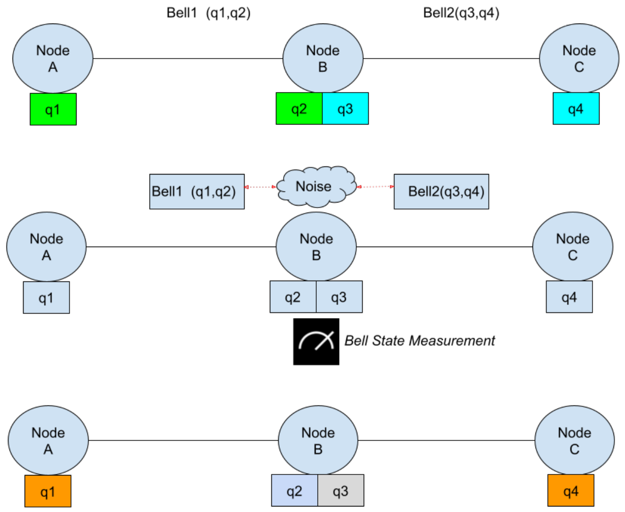


Figure 3.5: Set up for the entanglement swapping

In the set up of Figure 3.5 we can inspect that qubits q1 (node A) and q2 (node B) are entangled in Bell1(Bell state 1) and q4 (node C) and q3 (node B) are entangled in Bell2(Bell state 2). Node A and Node C, could create these entangled pairs and sent 1 qubit each into the intermediate Node B. The BSM(measurement) occurs in Node B, and as a consequence the initial entanglement between q3 and q4 qubits is destructed [12]. Nevertheless the state of q2 is "recreated"/reappears in the possition of qubit q4, which makes an new entanglement pair consisting of qubits (q1,q4).

Depending on the application requirements, the results of the BSM might be communicated to the Node C, in order to apply a correction operation on q4 depending on which Bell state was received. More specifically, the measurement collapses the quantum states of q2 and q3, but it also results in a change in the state of q1 and q4. Qubits q1 (Node A) and q4 (Node C) become entangled as a result of the BSM, but they are not measured themselves. This is known as the process of entanglement swapping. Now (q1,q4) share an entangled state, even though they were not directly entangled initially.

Fidelity calculates how close the state is to the desired state. As the noise in the system increases, the fidelity of the final state decreases, indicating the decrease in quantum entanglement quality.

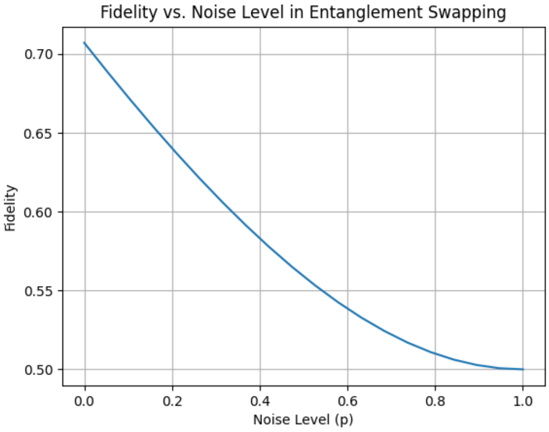


Figure 3.6: Fidelity vs Noise Level.

A value of 1 indicates the maximum perfect fidelity where there is no loss of entanglement, while lower values indicate a degradation due to noise. In practice a fidelity value less than 0.5 indicates that the state is unsuitable for further quantum processing. In Figure 3.6 we can inspect that as the noise level increases, the fidelity decreases, an expected behavior since noise disrupts the quantum states, making the final entangled state less similar to the ideal state.

3.4 Entanglement Swapping

For an entanglement swapping experiment, the IBM Quantum composer has been utilized for creating a circuit, which later was deployed in one of the available quantum processors. This experiment is actually a ”continuation” from the previous experiment presented in Figure 3.5. Again we start with 4 qubits (q[0], q[1],q[2] and q[3]) and the goal is to entangle q[0] and q[3]. In Figure 3.7 initially we create the entanglements alongside q[0] and q[1] using Hadamard gate in conjunction with a CNOT gate. Then the same procedure applies for qubits q[2] and q[3].

In Figure 3.8 we can observe on the left side of the screen the measurements of 1024 shots on the quantum processor(ibm_sherbrook) the circuit was deployed and as we were expecting we have the set of the observed values(vectorstates) 0000,0010,0010,0110 as results, stronly appearing qubits q[0] and q[3] entangled.

Using a different circuit for the same purpose shown in Figure 3.9, as provided by authors in [3], where also correction occurs conditionally based

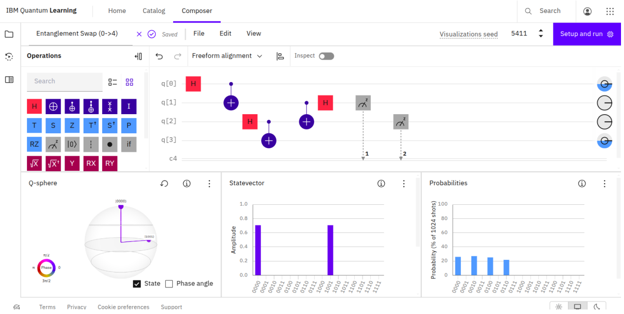


Figure 3.7: Design of the quantum circuit

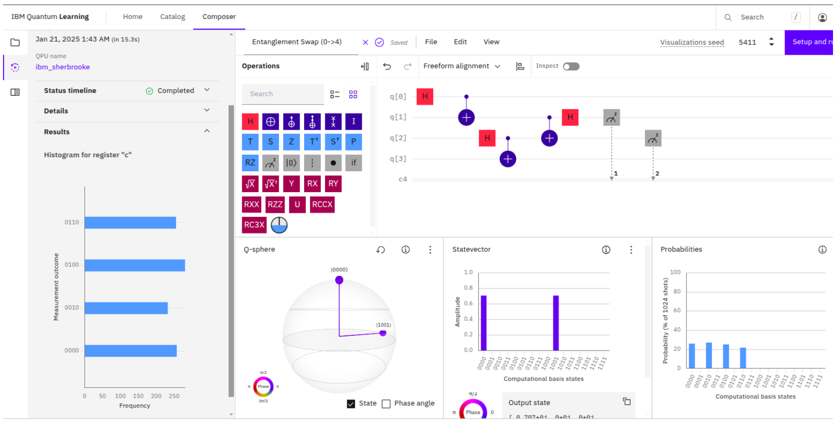
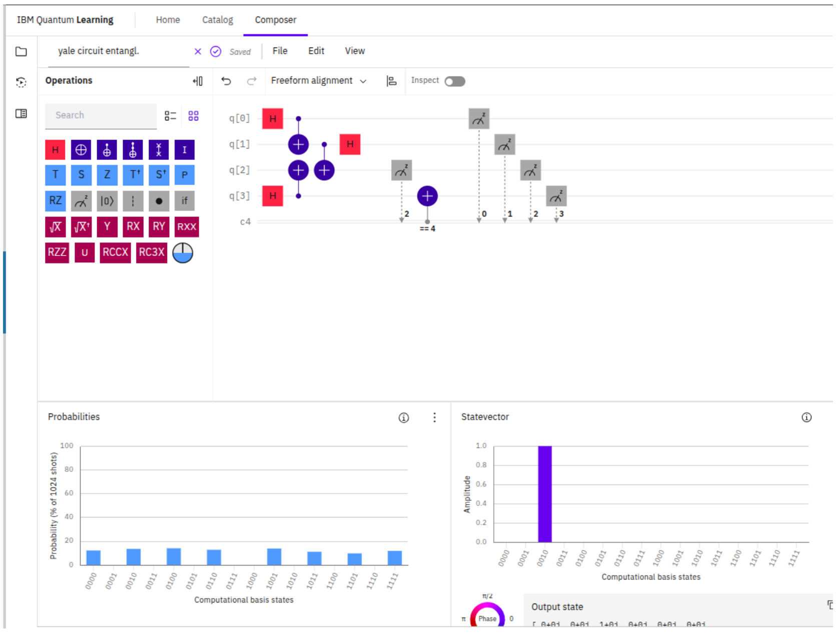


Figure 3.8: Left hand side of the screen depicts the results of measurement on the quantum processor

on the measured value of the qubit q[2], we can inspect again from the probabilities window, that the qubits q[0] and q[3] become eventually an entangled pair.

When we get the results from the IBM quantum processor though, in Figure 3.10, we can inspect that we have interference from factors that destabilize the ideal behavior of the quantum circuit, and we can see the presence of combinations that we were expecting not to be present on the measurement.

It is important to note(Figure 3.11) that even in a simpler experiment with just an entangled pair on real quantum processors might not perfect, and several factors can destabilize the expected states (like noise, gate imperfections, readout errors, and decoherence) are present and affecting the measurements. As we can inspect, on the probabilities only values (00 and 11) are expected, but from the real measurements on the left side of the screen we can observe mixed combinations shown up in a small ratio of the 1024 shots.

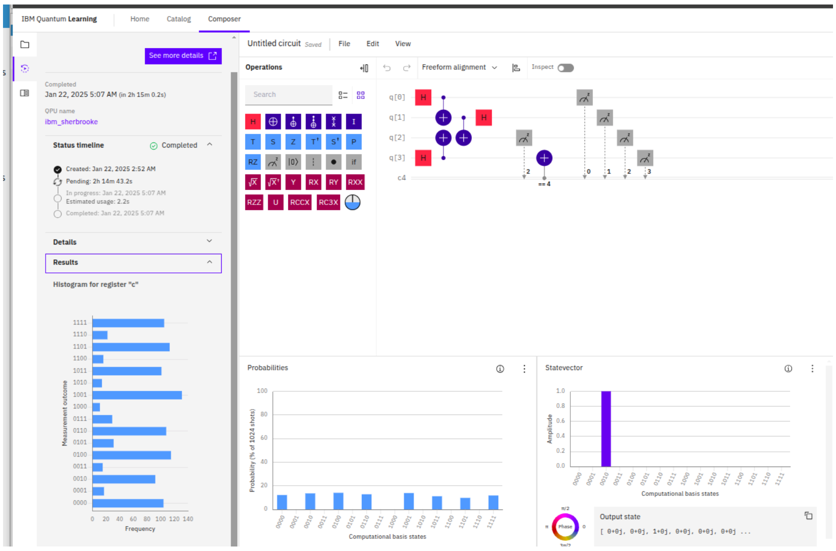


yale circuit entangl. Saved

Output state

```
[ 0+0j, 0+0j, 1+0j, 0+0j, 0+0j, 0+0j
```

Figure 3.9: Qubits q[0] and q[3] are an entangled pair



Untitled circuit Saved

Completed
Jan 22, 2025 5:07 AM (in 2h 15m 0.2s)

QPU name
ibm_sherbrooke

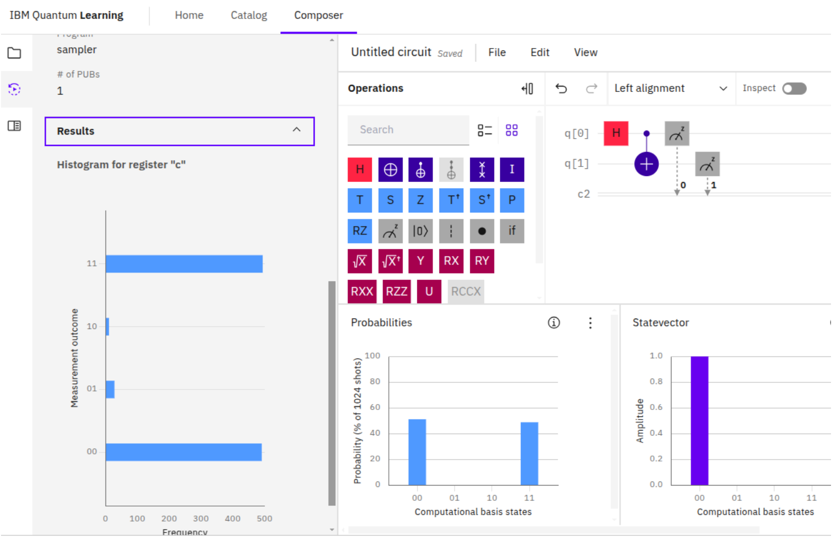
Status timeline: Completed

- Created: Jan 22, 2025 2:52 AM
- Pending: 2h 14m 43.2s
- In progress: Jan 22, 2025 5:07 AM
  Estimated usage: 2.2s
- Completed: Jan 22, 2025 5:07 AM

Output state

```
[ 0+0j, 0+0j, 1+0j, 0+0j, 0+0j, 0+0j ...
```

Figure 3.10: Results on the left hand side, with decoherence errors from running on a real quantum processor.



Program
sampler

# of PUBs
1

Figure 3.11: A simple entangled pair measurement on the quantum processor.

Chapter 4

Discussion

Chapter 5

Conclusion

Bibliography

[1] Khristo, Y., Asaad, B. and Abdelbaki, N. “Quantum Networking, where it is headed”, in *Proc. 16th Int. Computer Engineering Conf. (ICENCO)*, Cairo, Egypt, Dec. 2020, pp. 1–4.

[2] Van Meter, R., Satoh, R., Benchasattabuse, N., Teramoto, K., Matsuo, T., Hajdušek, M., Satoh, T., Nagayama, S. and Suzuki, S. ”A Quantum Internet Architecture”, *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2022, pp. 341-352.

[3] Valls, V., Promponas, P. and Tassiulas, L. ”A Brief Introduction to Quantum Network Control”, in *Comm. Mag.* 62, 10 (October 2024), pp. 48–53

[4] Benchasattabuse, N., Hajdušek, M. and Meter, R. V. ”Protocols for All-Photonic Quantum Repeaters” *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, Bellevue, WA, USA, 2023, pp. 314–315.

[5] Cacciapuoti, A. S., Caleffi, M., Tafuri, F., Cataliotti, F. S., Gherardini, S. and Bianchi, G. ”Quantum Internet: Networking Challenges in Distributed Quantum Computing,” in *IEEE Network*, vol. 34, no. 1, pp. 137-143, January/February 2020

[6] Pompili, M., Delle Donne, C., te Raa, I., van der Vecht, B., Skrzypczyk, M., Ferreira, G., de Kluijver, L., Stolk, A. J., Hermans, S. L. N., Pawełczak, P., Kozlowski, W., Hanson, R. and Wehner, S. (2022). Experimental demonstration of entanglement delivery using a quantum network stack. *npj Quantum Information*, *8*(1), 121.

[7] Khan, M. A., Aman, M. N. and Sikdar, B. ”Architecting the Quantum Future: Key Devices and Layers in Quantum Network Design” in *2024 IEEE Physical Assurance and Inspection of Electronics (PAINE)*, pp. 1-7

[8] Lv, Z., Cheng, C. and Song, H. ”Digital twins based on quantum networking”, in *IEEE Network*, 36(5),pp. 88-93 (2022)

[9] Li, Z., Xue, K., Li, J., Yu, N., Liu, J., Wei, L., Sun, Q. and Lu, J. ”Building a large-scale and wide-area quantum Internet based on an OSI-alike model”, in *China Communications*, vol. 18, no. 10, pp. 1–14, Oct. 2021.

[10] Kozlowski, W., Dahlberg, A. and Wehner, S. ”Designing a quantum network protocol”, in *Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies (CoNEXT 2020)*, Association for Computing Machinery, New York, NY, USA, 1–16.

[11] Cacciapuoti, A. S., Caleffi, M., Van Meter, R. and Hanzo, L. ”When Entanglement Meets Classical Communications: Quantum Teleportation for the Quantum Internet” in *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3808-3833, June 2020.

[12] Kozlowski, W., Wehner, S., Van Meter, R., Rijsman, B., Cacciapuoti, A. S., Caleffi, M. and Nagayama, S. (2023). *RFC 9340: Architectural Principles for a Quantum Internet.* RFC Editor. USA.

[13] Zhou, Y., Tang, Z., Nikmehr, N., Babahajiani, P., Feng, F., Wei, T. C., Zheng, H., and Zhang, P., ”Quantum computing in power systems,”*iEnergy*, 2022, vol 1, issue 2: 170-187

[14] Bhaskar, M.K., Riedinger, R., Machielse, B., Levonian, D. S., Nguyen, C. T., Knall, E. N., Park, H., Englund, D., Lončar, M., Sukachev, D. D.i and Lukin, M. D. ”Experimental demonstration of memory-enhanced quantum communication” in *Nature*, 580, pp 60–64 (2020).

[15] Urbanek, M., Nachman, B., Pascuzzi, V. R., He, A., Bauer, C. W., and de Jong, W. A., ”Mitigating Depolarizing Noise on Quantum Computers with Noise-Estimation Circuits,” *Phys. Rev. Lett.*, vol. 127, no. 27, p. 270502, Dec. 2021.

[16] Caleffi, M. and Cacciapuoti, A. S. ”Quantum Switch for the Quantum Internet: Noiseless Communications Through Noisy Channels,” in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 3, pp. 575-588, March 2020

[17] Alshowkan, M., Evans, P., Williams, B., Rao, N., Marvinney, C., Pai, Y., Lawrie, B., Peters, N., and Lukens, J. ”Advanced architectures for high-performance quantum networking”, in *Journal of Optical Communications and Networking*, 14, pp. 493-499.(2022)

[18] Bennett, C. H and Brassard, G. "Quantum cryptography: Public key distribution and coin tossing," in *Proc. IEEE Int. Conf. Computers, Systems and Signal Processing*, 1984, pp. 175–179.

[19] King, C. "The capacity of the quantum depolarizing channel" in *Information Theory, IEEE Transactions* on. 49. pp. 221 - 229 (2003).

[20] Wallnöfer, J., Pirker, A., Zwerger, M. and Dür, W. "Multipartite state generation in quantum networks with optimal scaling", in *Scientific Reports*, vol. 9, no. 1, article 314, Jan. 2019.

[21] Johansson, J. R., Nation, P. D. and Nori, F. *"QuTiP 2: A Python framework for the dynamics of open quantum systems", Comp. Phys. Comm. 184*, 1234 (2013).

[22] Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C. J., Lishman, J., Gacon, J., Martiel, S., Nation, P., Bishop, L. S., Cross, A. W., Johnson, B. R. and Gambetta, J. M. (2024, May 15). "Quantum computing with Qiskit" arXiv. https://arxiv.org/abs/2405.08810. Accessed January 20, 2025.

[23] Salek, S., Ebler, D. and Chiribella, G., "Quantum communication in a superposition of causal orders", arXiv preprint arXiv:1809.06655. Accessed January 21, 2025.

Appendix A: Simulation Code

Listing 1: BB84 for QKD

```
from qutip import basis, ket2dm
import numpy as np
zero = basis(2, 0)  ''' |0> '''
one = basis(2, 1)   ''' |1> '''
plus = (zero + one).unit()  ''' |+> '''
minus = (zero - one).unit() ''' |-> '''
''' Alice's qubits'''
def generate_bb84_states(num_qubits):
    states = []
    bases = []
    for _ in range(num_qubits):
        basis_choice = np.random.choice(['z', 'x'])
        bit = np.random.choice([0, 1])
        if basis_choice == 'z':
            states.append(zero if bit == 0 else one)
        else:
            states.append(plus if bit == 0 else minus)
        bases.append(basis_choice)
    return states, bases
'''Measurement'''
def measure_state(state, basis):
    if basis == 'z':
        projection = [zero, one]
    else:  ''' 'x' '''
        projection = [plus, minus]
    probabilities = [abs(proj.overlap(state))**2 for proj in projection]
    return np.random.choice([0, 1], p=probabilities)
'''Simulation'''
num_qubits = 100
alice_states, alice_bases = generate_bb84_states(num_qubits)
bob_bases = np.random.choice(['z', 'x'], num_qubits)
''' Measure and compare '''
bob_results = [measure_state(state, bob_bases[i]) for i, state in enumerate(
    ↪ alice_states)]
matching_bases = [alice_bases[i] == bob_bases[i] for i in range(num_qubits)]
shared_key = [bob_results[i] for i in range(num_qubits) if matching_bases[i
    ↪ ]]
''' Calculate success rate'''
success_rate = len(shared_key) / num_qubits
print(f"Key␣Exchange␣Success␣Rate:␣{success\_rate:.2f}")
import matplotlib.pyplot as plt
''' Visualization 1: Basis Matching (Bar Chart)'''
matched = sum(matching_bases)
mismatched = num_qubits - matched
plt.figure(figsize=(8, 6))
plt.bar(['Matched', 'Mismatched'], [matched, mismatched], color=['green', '
    ↪ red'])
plt.title('Basis␣Matching␣in␣BB84␣Protocol')
plt.xlabel('Basis␣Match')
plt.ylabel('Number␣of␣Qubits')
plt.savefig('Basis␣Matching␣in␣BB84␣Protocol.png')
''' Visualization 2: Success Rate vs. Number of Qubits (Line Chart) '''
num_qubits_list = [10, 50, 100, 200, 500, 1000, 2000, 3000, 4000, 5000, 8000
    ↪ , 10000]
success_rates = []
for nq in num_qubits_list:
    alice_states, alice_bases = generate_bb84_states(nq)
    bob_bases = np.random.choice(['z', 'x'], nq)
    bob_results = [measure_state(state, bob_bases[i]) for i, state in
        ↪ enumerate(alice_states)]
    matching_bases = [alice_bases[i] == bob_bases[i] for i in range(nq)]
    shared_key = [bob_results[i] for i in range(nq) if matching_bases[i]]
    success_rate = len(shared_key) / nq
    success_rates.append(success_rate)
plt.figure(figsize=(8, 6))
plt.plot(num_qubits_list, success_rates, marker='o', color='b')
plt.title('Success␣Rate␣vs.␣Number␣of␣Qubits')
plt.xlabel('Number␣of␣Qubits')
plt.ylabel('Success␣Rate')
plt.grid(True)
plt.savefig("Success␣Rate␣vs␣Number␣of␣Qubits.png")
```

Listing 2: Entanglement swapping(fidelity vs polarizing noise)

```
from qutip import *
import numpy as np
import matplotlib.pyplot as plt

''' Define Bell state '''
def create_bell_state():
    zero = basis(2, 0)
    one = basis(2, 1)
    bell_state = (tensor(zero, zero) + tensor(one, one)).unit()
    bell_dm = ket2dm(bell_state)  ''' Convert to density matrix, shape: (4,
        ↪ 4)'''
    return bell_dm

''' Apply noise to a quantum state (Depolarizing channel)'''
def apply_depolarizing_noise(state, p):
    dim = np.prod(state.dims[0])  ''' Total dimension of the system'''
    identity = qeye(dim)  ''' Identity operator, shape: (dim, dim)'''
    identity.dims = state.dims
    n1=(1-p)*state
    n2 = (p/dim)*identity
    noisy_state = (1 - p) * state + (p / dim) * identity
    return noisy_state

''' Entanglement swapping '''
def entanglement_swapping(bell1, bell2):
    """
    Perform entanglement swapping between two Bell states:
    bell1: A-B (shape: (4, 4)), bell2: B-C (shape: (4, 4))
    Returns:
    - swapped_state: final entangled state between A and C (shape: (4, 4))
    """
    # Bell measurement on qubits 2 and 3 (middle qubits)
    bell_meas = (tensor(basis(2, 0), basis(2, 0)).proj() +
                 tensor(basis(2, 1), basis(2, 1)).proj())  # Bell basis,
                     ↪ shape: (4, 4)
    projection = tensor(qeye(2), bell_meas, qeye(2))  # Operate on qubits 2
        ↪ and 3, shape: (16, 16)

    # Combine the two input states into a joint density matrix
    joint_state = tensor(bell1, bell2)  # Combined state, shape: (16, 16)

    # Apply projection for entanglement swapping
    swapped_state = (projection * joint_state * projection.dag()).unit()  #
        ↪ Shape: (16, 16)
    # Partial trace over qubits 2 and 3 to get A-C state
    final_state = swapped_state.ptrace([0, 3])  # Shape: (4, 4)
    return final_state

# Network Simulation
def simulate_network():
    """
    Simulates a basic quantum repeater network with entanglement swapping.
    Returns:
    - fidelity_value: fidelity of the final entangled state with an ideal
        ↪ Bell state
    """
    noise_levels = np.linspace(0, 1, 20)  # Noise levels from 0 to 1
    fidelities = []


    ''' Step 1: Create initial entanglement between nodes'''
    bell1 = create_bell_state()  # Between nodes A and B
    print(bell1)

    bell2 = create_bell_state()  # Between nodes B and C
    print(bell2)

    ''' Step 2: Calculate fidelity for different noise levels '''
    for p in noise_levels:
        bell1_noisy = apply_depolarizing_noise(bell1, p)
        bell2_noisy = apply_depolarizing_noise(bell2, p)
        final_state = entanglement_swapping(bell1_noisy, bell2_noisy)
        ideal_bell = create_bell_state()
        print(ideal_bell)
        fidelity_value = fidelity(final_state, ideal_bell)
        fidelities.append(fidelity_value)

    ''' Step 3: Plot fidelity vs noise level'''
    plt.plot(noise_levels, fidelities)
    plt.xlabel("Noise␣Level␣(p)")
    plt.ylabel("Fidelity")
    plt.title("Fidelity␣vs.␣Noise␣Level␣in␣Entanglement␣Swapping")
    plt.grid(True)
    plt.savefig("Fidelity␣vs␣Noise␣Level␣in␣Entanglement␣Swapping.png")

# Main Execution
if __name__ == "__main__":
    simulate_network()
```

Listing 3: QKD-AES application

```
from qutip import basis, ket2dm
import numpy as np
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
import os

# Define basis states of computational basis and Hadamard basis
zero = basis(2, 0)   # |0>
one = basis(2, 1)    # |1>
plus = (zero + one).unit()  # |+>
minus = (zero - one).unit()  # |->

# Alice's qubits
```

```
def generate_bb84_states(num_qubits):
    states = []
    bases = []
    for _ in range(num_qubits):
        basis_choice = np.random.choice([’z’, ’x’])
        bit = np.random.choice([0, 1])
        if basis_choice == ’z’:
            states.append(zero if bit == 0 else one)
        else:
            states.append(plus if bit == 0 else minus)
        bases.append(basis_choice)
    return states, bases

# Measurement
def measure_state(state, basis):
    if basis == ’z’:
        projection = [zero, one]
    else:  # ’x’
        projection = [plus, minus]
    probabilities = [abs(proj.overlap(state))**2 for proj in projection]
    return np.random.choice([0, 1], p=probabilities)

# Generate shared key
def generate_shared_key(num_qubits):
    alice_states, alice_bases = generate_bb84_states(num_qubits)
    bob_bases = np.random.choice([’z’, ’x’], num_qubits)

    # Bob measures Alice’s qubits
    bob_results = [measure_state(state, bob_bases[i]) for i, state in
        ↪ enumerate(alice_states)]
    matching_bases = [alice_bases[i] == bob_bases[i] for i in range(num_
        ↪ qubits)]
    shared_key = [bob_results[i] for i in range(num_qubits) if matching_
        ↪ bases[i]]

    # Success rate
    success_rate = len(shared_key) / num_qubits
    print(f"Key␣Exchange␣Success␣Rate:␣{success_rate:.2f}")

    # Ensure we have a sufficiently long key for AES (128 bits = 16 bytes)
    if len(shared_key) >= 16:
        shared_key = shared_key[:16]
        return bytes(shared_key), success_rate
    else:
        print("Not␣enough␣key␣bits␣generated.␣Increase␣number␣of␣qubits.")
        return None, success_rate

# Encrypt and decrypt messages using AES
def encrypt_message(message, key):
    cipher = AES.new(key, AES.MODE_CBC)  # Use AES in CBC mode
    iv = cipher.iv
    encrypted_message = cipher.encrypt(pad(message.encode(), AES.block_size)
        ↪ )
    return iv, encrypted_message

def decrypt_message(iv, encrypted_message, key):
    cipher = AES.new(key, AES.MODE_CBC, iv=iv)
    decrypted_message = unpad(cipher.decrypt(encrypted_message), AES.block_
        ↪ size)
    return decrypted_message.decode()

# Main simulation
```

```
num_qubits = 256
success_rate = 0
RATE = 0.57
while(success_rate <= RATE):
    key, success_rate = generate_shared_key(num_qubits)
    if(success_rate < RATE):
        print("retry␣key␣generation")

if key and success_rate > RATE:  # Use the key if success rate is high
    ↪ enough
    print("Shared␣key␣established:", key)

    # Alice sends a message to Bob
    alice_message = "Hello␣Bob!"
    iv, encrypted_message = encrypt_message(alice_message, key)
    print(f"Alice's␣encrypted␣message:␣{encrypted_message}")

    # Bob decrypts the message
    bob_message = decrypt_message(iv, encrypted_message, key)
    print(f"Bob's␣decrypted␣message:␣{bob_message}")

    # Bob sends a message to Alice
    bob_message = "Hello␣Alice!"
    iv, encrypted_message = encrypt_message(bob_message, key)
    print(f"Bob's␣encrypted␣message:␣{encrypted_message}")

    # Alice decrypts the message
    alice_message = decrypt_message(iv, encrypted_message, key)
    print(f"Alice's␣decrypted␣message:␣{alice_message}")
else:
    print("Failed␣to␣generate␣a␣secure␣shared␣key.")
```