Analysis of the NS-3 module QKDNetSim for the Simulation of QKD Networks

David Soler, Iván Cillero, Francisco J. Nóvoa, Carlos Dafonte, Ana Fernández Vilas, and Manuel Fernández-Veiga

Laboratorio Interdisciplinar de Aplicaciones de Inteligencia Artificial, Faculty of Computer Science, Universidade da Coruña, 15071 A Coruña, Spain
Laboratorio de Telemática, Faculty of Computer Science, Universidade da Coruña, 15071 A Coruña, Spain
I&CLabs, *AtlanTTic*, University of Vigo, 36310 Vigo, Spain
Correspondence: david.soler@udc.es

DOI: *https://doi.org/10.17979/spudc.000024.20*

Abstract: Quantum Key Distribution (QKD) is a promising technology that allows two nodes to privately agree on a key through a quantum channel. Unfortunately, QKD is still in experimental phase and researchers must rely on simulators to replicate the behaviour of a quantum network. One of the most widespread is *QKDNetSim*, a module for the C++ network simulator *NS-3*. However, this module is very limited in its behaviour, so it does not faithfully represent a real quantum network. In this work we analyse the structure and components of *QKDNetSim*, as well as its shortcomings and how they affect the quality of the simulation.

1 Introduction

Quantum technologies have experienced a significant advance in recent years Cao et al. (2022); Illiano et al. (2022); Singh et al. (2021). The increasing computing power of quantum computers is endangering the cryptographic algorithms that are employed to distribute keys between users, including protocols as widespread as HTTPS. Nevertheless, quantum technologies also provide an alternative to these algorithms: the Quantum Key Distribution (QKD) protocols allow two nodes to agree on a key through a quantum channel, in such a way that it would be impossible for an eavesdropper to obtain the key without being detected. This key can then be used to encrypt communications between the two nodes.

The number of implemented QKD networks is currently very small due to the high cost of the required material and the lack of maturity in the technology. Thus, researchers must employ simulators that imitate the behavior of a quantum network. There are multiple alternatives depending on the scope of the research Aji et al. (2021): some simulators focus on representing the physical layer of the quantum channel, while others allow users to define entire networks in which nodes can execute QKD between them. The simulators "Qunetsim" and "NetSquid", both written for Python, are among the most popular options.

The network simulator NS-3 is widely used in the scientific and educational communities due to its level of detail and its customizing capabilities. There exists a module implemented for NS-3 for the simulation of quantum networks, named QKDNetSim Mehic et al. (2017), that was developed by researchers of the Technical University of Ostrava.

The advantages of QKDNetSim over other simulators come from the granularity of NS-3: this simulator allows for in-depth configuration of every component, and packets sent over the simulated network are fully defined, including headers for all protocols involved. This level of

detail, when applied to QKDNetSim's simulation of quantum networks, could help newcomers to understand the fundamentals of quantum communications. However, this module contains numerous conceptual and implementation flaws that affect its ability to faithfully represent the behavior of a real quantum network.

In this work we evaluate the quantum network simulation module QKDNetSim: of which elements it is composed, what procedures are followed to simulate the key exchange and the encryption of information, and to which degree it imitates the behavior of a real quantum network. We will also analyze the limitations and errors that prevent QKDNetSim from achieving its purposed objectives.

The rest of this work is organised as follows: in Section 2 we introduce readers to the basics of Quantum Key Distribution, which will be required for understanding the components and behaviour of the simulator. In Section 3 we analyse the module QKDNetSim. First, we introduce the components of a node connected to a quantum channel. Then, we comment on QKDNetSim's errors and how they affect its usefulness. Finally, Section 4 will conclude this document and discuss future improvements.

2 Background

2.1 Quantum key distribution

Quantum channels are communication channels that are analogous to classical channels, but transmit *qubits* instead of bits. Qubits represent a superposition of their base states $|0\rangle$ and $|1\rangle$ in the following way:

$$|\Phi\rangle = \alpha|0\rangle + \beta|1\rangle$$

where α and β represent the relative probability of measuring each of the base states.

Unlike classical bits, qubits possess two properties that make them specially suited for transmitting private information: the **no-cloning theorem** (it is impossible to perfectly clone a quantum state) and the **uncertainty principle** (whenever a state is measured, it collapses into a base state and loses its superposition). Taken together, these two properties ensure that any attacker trying to eavesdrop a quantum channel will inevitably modify the qubits that are being transmitted, such that the legitimate interlocutors will be aware of the intrusion.

Since quantum channels are capable of transmitting information in a private manner, they can be used to establish common cryptographic material between two parties. This is called Quantum Key Distribution (QKD), and it is believed to be information-theoretically secure, that is, it would be impossible for an attacker to obtain the secret no matter how many resources they have access to. The shared secret can then be used in any classical cryptographic algorithm that is considered secure, like AES.

Figure 1 shows the structure of a QKD communication: Peers Alice and Bob are connected through both classical and quantum channels. They must first establish a common secret executing a QKD protocol like BB84 Bennett and Brassard (1984). If they detect any eavesdropping, they abort the protocol. Then, Alice uses this common secret as a symmetric key to encrypt a message which she transmits to Bob. Finally, he decrypts the message.

As mentioned, the main advantage of QKD over other key distribution protocols is that QKD is information-theoretically secure. While classical algorithms like Diffie-Hellman or RSA base their security on the complexity of concrete mathematical problems, QKD's security is guaranteed by the laws of physics.

2.2 QKD network architecture

In experimental settings, the execution of QKD is very slow and thus it would be very inefficient to block the generation of a packet to obtain key material through QKD Mehic et al. (2020). For that reason, it is common to asynchronously execute the QKD protocol at a previous time, and

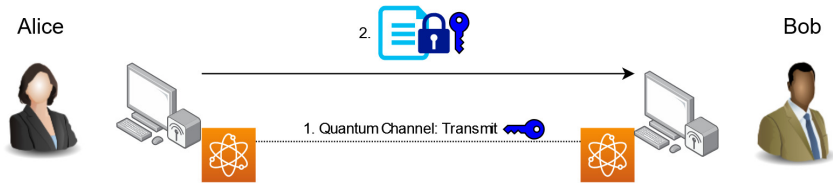

Figure 1: Diagram of a QKD communication.

store the resulting key material in a buffer to be consumed when needed Cao et al. (2019). Thus, the processes of generating and consuming key material are decoupled.

However, this introduces the need to synchronise the buffers of adjacent nodes. If Alice encrypts a packet using material from her buffer, Bob should be able to obtain the exact same key from his. If an error occurs and the buffers are desynchronised, the obtained keys would not match and Bob would not be able to decrypt Alice's messages.

3 Analysis of QKDNetSim

3.1 NS-3

NS-3 is a network simulator written in C++ that is widely used by students and academics. It provides an environment for the simulation of events in a computer network, as well as classes that can generate said events. It is open source, and there are multiple modules developed by the community, including QKDNetSim, the object of study of this work.

NS-3 provides classes that represent certain components of real-world networks. *Nodes* represent the computers inside the network, and they are connected to each other through *NetDevices*. *Applications* simulate the execution of programs inside Nodes, and they generate *Packets*. *Helpers* are classes that simplify the process of modifying Nodes, usually installing a NetDevice or Application.

The user can determine the duration of the simulation, as well as the timing of every event (when the Applications start sending Packets and how often). When it ends, it is possible to generate a PCAP file listing all packets to analyse the simulation.

3.2 Structure of a QKDNetSim node

The module QKDNetSim adds to NS-3 the possibility of creating quantum channels between pairs of nodes. To this end, the following components are added to each pair of nodes:

- **Send/Receive Applications**: they simulate programs that create packets to be encrypted by the rest of QKDNetSim's components. When they are generated or received, they are sent to the Manager to perform the pertinent cryptographic operations.
- **QKD Manager**: the central component of the module, which serves as connection between the others. The Manager processes incoming and outgoing packets, identifies which operations need to be performed and calls the pertinent component to execute them.
- **Cryptography Handler**: receives petitions from the Manager to encrypt or decrypt packets. It implements multiple cryptographic algorithms, and has access to the Key Buffer.
- **Simulated Quantum Channel**: imitates the behaviour of a quantum channel. Each of the nodes possesses a *Charging Application*, which constantly generate new shared key material.
- **Key Buffer**: Stores key material transmitted through the quantum channel for future use.

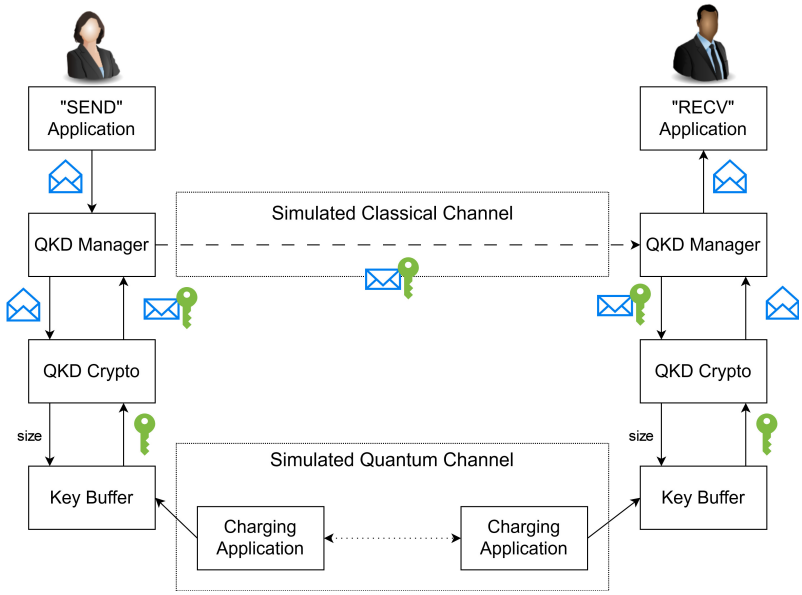

Figure 2: Architecture of QKDNetSim.

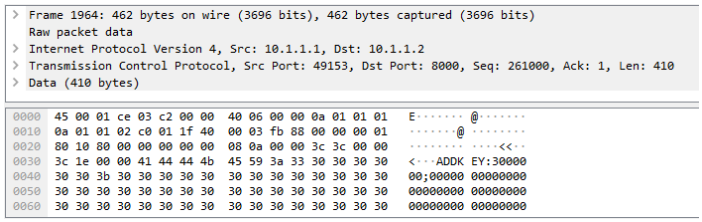

Figure 3: Packet containing key material sent by Charging Application.

The interaction between elements is shown in Figure 2.

3.3 Simulation of a Quantum Channel

As mentioned, the quantum channel is simulated through the *Charging Applications* that are installed for each pair of connected nodes. Figure 3 shows the contents of a packet exchanged between the Charging Applications of two adjacent nodes, which includes (after the label *ADDKEY*) the key material that will be added to the buffers.

QKDNetSim uses buffers to store key material, as explained in Section 2.2. the Charging Applications are constantly generating new key material and storing it into their respective Key Buffers. Whenever the Send Application generates a new packet, some of the previously generated key material is consumed to encrypt it.

Figure 4 shows the amount of key material stored inside a Key Buffer during a simulation. It increases periodically when the Charging Applications creates new key material, and it decreases when new packets are encrypted.

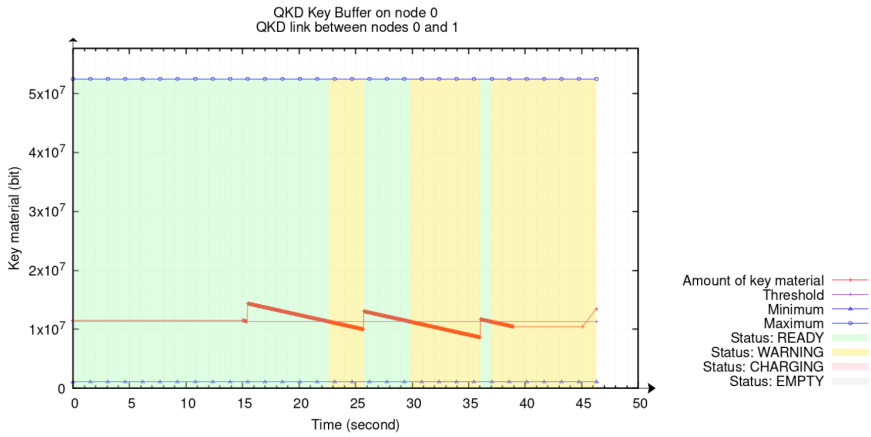

Figure 4: Amount of available key material inside the Key Buffer throughout the simulation.

3.4 Shortcomings

The objective of QKDNetSim is to faithfully represent the behaviour of a real quantum network. However, some of its components have been simplified to a point that they do not longer work as their real counterparts would. Instead of making the simulation easier to understand, these simplifications limit the usefulness of QKDNetSim. Furthermore, there are implementation errors that render some components unusable.

We highlight the following shortcomings of QKDNetSim:

1. The messages sent through the Simulated Quantum Channel are all identical. The Simulated Quantum Channel is tasked with the creation of new key material between two nodes. However, all messages exchanged between the Charging Applications of adjacent nodes contain only a string of '0's, as can be seen in Figure 3. Since all the key material that is created is the same, all packets are encrypted with the same key. Thus, in QKDNetSim the nodes are not actually executing a Key Distribution protocol, since they already know that the key is a string of '0's. This defeats the purpose of QKD and makes QKDNetSim a much less realistic simulator.

2. The Key Buffers do not store keys. The key material generated through the Simulated Quantum Channel is not stored anywhere. This should be the function of the Key Buffer: instead, it only stores a number which represents the *amount* of key material it should have.

Whenever the Simulated Quantum Channel tries to insert new key material in the Key Buffer, it is discarded and the *amount* is increased. A similar process occurs when the Manager requests key material from the Buffer: a new string of '0's is returned and the *amount* is decreased by its length. The plot shown in Figure 4 represents this *amount*, since is the only metric the Buffer can provide.

Since there is no real key material inside the Buffers, there is also no mechanism for ensuring, maintaining or recovering synchronicity between Key Buffers of connected nodes. This makes QKDNetSim less realistic, as it does not represent a problem that needs to be addressed in real quantum networks.

3. Encryption is disabled and does not work. The Cryptography Handler provides implementations for the encryption algorithms AES and One-Time Pad (OTP). However, these

implementations contain errors: they incorrectly assume keys are represented as an array of bytes, when they are an array of bits. Thus, whenever the simulator tries to encrypt a packet, it crashes.

Even if these errors are corrected and packets are adequately encrypted, the PCAP still shows them in plaintext. This is due to the order in which QKDNetSim performs its operations: packets are logged and inserted into the PCAP before being encrypted. The fact that contents of the PCAP do not correspond to the real messages that were exchanged makes the simulation harder to understand.

4 Conclusion

In this work, we have analysed the Quantum Key Distribution module of the simulator NS-3, called QKDNetSim. The module defines a set of components that work together replicate a quantum channel between two nodes. The structure of QKDNetSim is very promising, since its each of its components is clearly defined, performs a very specific action and adequately represents an element that exists in real quantum networks.

However, some of this components are only surface-deep, and they usually do not perform the actions that they are supposed to. Correcting the shortcomings explained in this work would greatly improve the quality of QKDNetSim, such that it could represent a real quantum network in a more faithful way. Furthermore, these modifications do not require substantial alteration to the foundation of the module, thus simplifying the process of improving QKDNetSim.

Acknowledgements

This work is part of the project TED2021-130369B-C31, TED2021-130369BC32, TED2021-130369B-C33 and TED2021-130492B-C21 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

The work is also funded by the Plan Complementario de Comunicaciones Cuánticas, Spanish Ministry of Science and Innovation(MICINN), Plan de Recuperación NextGenerationEU de la Unión Europea (PRTR-C17.I1, CITIC Ref. 305.2022), and Regional Government of Galicia (Agencia Gallega de Innovación, GAIN, CITIC Ref. 306.2022)

Bibliography

A. Aji, K. Jain, and P. Krishnan. A survey of quantum key distribution (qkd) network simulation platforms. In *2021 2nd Global Conference for Advancement in Technology (GCAT)*, pages 1–8, 2021.

C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7–11, dec 1984.

Y. Cao, Y. Zhao, J. Wang, X. Yu, Z. Ma, and J. Zhang. Kaas: Key as a service over quantum key distribution integrated optical networks. *IEEE Communications Magazine*, 57(5):152–159, 2019.

Y. Cao, Y. Zhao, Q. Wang, J. Zhang, S. X. Ng, and L. Hanzo. The evolution of quantum key distribution networks: On the road to the qinternet. *IEEE Communications Surveys & Tutorials*, 24(2):839–894, 2022.

J. Illiano, M. Caleffi, A. Manzalini, and A. S. Cacciapuoti. Quantum internet protocol stack: A comprehensive survey. *Computer Networks*, 213:109092, aug 2022.

M. Mehic, O. Maurhart, S. Rass, and M. Voznak. Implementation of quantum key distribution network simulation module in the network simulator ns-3. *Quantum Information Processing*, 16(10):1–23, oct 2017.

M. Mehic, M. Niemiec, S. Rass, J. Ma, M. Peev, A. Aguado, V. Martin, S. Schauer, A. Poppe, C. Pacher, and M. Voznak. Quantum key distribution: A networking perspective. *ACM Computing Surveys*, 53(5), sep 2020.

A. Singh, K. Dev, H. Siljak, H. D. Joshi, and M. Magarini. Quantum internet—applications, functionalities, enabling technologies, challenges, and research directions. *IEEE Communications Surveys & Tutorials*, 23(4):2218–2247, 2021.