Systematic Analysis of Quantum Key Distribution Network Simulations Tools and Platforms

Ivan Cvitić(✉), Dragan Peraković, and Josip Vladava

Faculty of Transport and Traffic Sciences, University of Zagreb, Vukelićeva 4, 10000 Zagreb, Croatia
{ivan.cvitic,dragan.perakovic,josip.vladava}@fpz.unizg.hr

Abstract. Quantum Key Distribution (QKD) is one of the applications of quantum communications that is used to generate and distribute secret keys between two parties. These secrete keys are then used to establish secure communication. The security of QKD relies upon the principles of quantum mechanics, such as no-cloning theorem, that allow detection of any disturbances that quantum states could experience during transit. To facilitate these kinds of applications requires the development of quantum networks that consist of specialized hardware that is more complex than its classical counterpart. Hardware used in quantum networks is expensive, therefore it is very important to be confident in the design of network before physical implementation. This is why the development of quantum network simulation tools is of utmost importance. This research paper gives an overview and analysis of quantum network simulation tools and platforms that are capable of simulating quantum key distribution. Analysis will result in characteristics and capabilities of simulation tools and platforms that the researchers can use to choose the simulation tools most suitable for their scenario.

Keywords: quantum key distribution · quantum communication network simulation · QKD · quantum entanglement

1 Introduction

Quantum communications are gaining increasing importance and finding a growing number of applications. Quantum Key Distribution (QKD) is one such application that offers the possibility of enhancing communication security through the exchange of quantum keys based on the laws of quantum mechanics. This approach significantly increases the security of existing (classical) communications by mitigating known threats targeting the exchange of cryptographic keys, digital certificates, and the overall public key infrastructure. Furthermore, this concept of key exchange and creation provides resilience against future threats posed by the development of quantum computers to existing key exchange methods.

D. Perakovic and L. Knapcikova (Eds.): FABULOUS 2023, LNICST 542, pp. 3–17, 2024.
https://doi.org/10.1007/978-3-031-50051-0_1

Currently, high costs of building blocks in a quantum network, which consist of devices such as Entangled Photon Source (EPS), Superconducting Nanowire Single-photon Detector (SNSPD), Single-photon Avalanche Detector (SPAD) and dark optical fibers, as well as the high expenses associated with the physical implementation of such network represents challenges in physical implementation. To address those challenges, simulations of such networks are of great importance as they provide insights into the events and processes occurring within the network. By exploring different sets of input parameters, simulations allow us to understand their impact on the functioning of the network as a whole. These results have the potential to serve as a foundation and reference for further research and decision-making regarding the feasibility of physically implementing a quantum communication network. Despite the clear significance of simulations in the field of quantum communications, there is a noticeable lack of research and analysis on software platforms and tools designed for such narrow purpose. This lack of research is further exacerbated in the area of network simulations for quantum key distribution, which is a subset of quantum communications. The reason for this is the extremely limited number of research groups working on this and similar research problems.

As the research field of quantum communications and QKD increasingly gains significance and demonstrates growing applications, the need for simulating events and processes occurring in such networks becomes more apparent. This highlights the necessity of synthesizing knowledge about existing simulation tools and platforms, as well as identifying their capabilities, characteristics, functionalities, and observed limitations. The results of this research will provide support to researchers in selecting appropriate simulation tools for future investigations. Additionally, they will underscore the need and importance of enhancing existing tools in terms of functionality, capabilities, and applicability across different scenarios.

The structure of this paper is organized as follows. Section 2 presents previous research on quantum network simulators. These research papers consist of relevant simulation tools, and comparison based information. The simulators were compared based on the programming language in which they were written, are they open sourced, do they have implemented noise models, are they modular, and do they have implemented QKD protocols. Section 3 provides an explanation of quantum key distribution (QKD) and describes the architecture of QKD networks. The architecture of these networks consists of quantum links, classical links and quantum nodes as elementary components that enable the transmission of entangled photons and accompanying classical information. Section 4 provides an overview and general information on the available simulation platforms for quantum networks. Section 5 presents a detailed analysis of simulators in the context of simulating QKD networks. This section compares simulators based on functionality and capabilities, such as: error modelling, programming interface, size of simulated network, node types, etc.

2 Previous Research

Researchers in [1] presents a survey of the latest, at the time of writing, simulation frameworks which are used to simulate different types of QKD protocols and QKD quantum networks. These simulation frameworks are used to develop and implement efficient and

practical quantum cryptographic schemes and network protocols. This paper contains comparisons of the simulation frameworks based on a number of characteristics. Authors performed comprehensive, systematic analysis of experimental simulation frameworks and their functionalities with the ultimate goal of paving the way for a future, universal quantum testbed. Authors analyzed simulation frameworks designed to simulate only the QKD protocols and frameworks designed to simulate QKD networks and protocols within that network. Table 1 presents a summary of the main characteristics and functionalities of researched simulation frameworks.

Modularity was described as one of the most important characteristics of a QKD simulator, because it makes the development and modeling of proprietary networks and protocols easier [1]. Modularity enables the developer to take already implemented functionalities and change them or build on them.

Table 1. Comparison of characteristics of QKD simulation frameworks [1]

Simulators	Simulation Environment	Supported protocols	Modularity	Available publicly
QuCCs [2]	Java, Matlab, MySQL	BB84, B92	Yes	No
qkdSim [3]	Python	B92	Yes	No
EnQuad [4]	Matlab	BB84	Yes	Yes
Fan-Yuan et al. [5]	C++, OMNet++	BB84, MDI-QKD	No	No
Kreinberg et al. [6]	VPItransmissionMaker OpticalSystems	Weak Coherent Prepare-and_Measure CV-QKD	No	No
SeQUeNCe [7]	N/A	BB84, Cascade, Teleportation Based Protocols	Yes	Yes
QuNetSim [8]	Python	All generic Protocols	No	Yes
NetSquid [9]	Python	All generic protocols	Yes	Yes
Wu et al. [10]	N/A	BB84	No	No

With this paper, authors emphasized the importance of continuous development of quantum network simulators, since the practical deployment of these networks is still complex and expensive. The ideal simulator was described as modular and universal [1]. Research [11] is mainly focused on distributed quantum computing. This paper provides an overview of the main challenge and problems of distributed quantum computing. Distributed quantum computing will be enabled with the implementation of Quantum Internet, as such authors have provided a detailed analysis of simulation tools that are used to simulate quantum networks. Analysis is done from the perspective of distributed quantum computing, and tools are categorized within three classes: hardware-oriented, protocol-oriented and application-oriented.

Table 2 consists of distributed quantum communication simulation tools, their characteristics and categorization according to classes. These simulation tools can also be used to simulate quantum networks.

Table 2. Comparison of distributed quantum communication simulation tools [11]

Simulation tool	Language	Noise models	Open source	Class
SQUANCH	Python	Yes	Yes	HW
NetSquid	Python	Yes	No	HW
SimulaQron	Python	No	Yes	PR
SeQUeNCe	C++ /Python	Yes	Yes	PR
QuISP	C++	Yes	Yes	PR
QuNetSim	Python	No	Yes	PR
NetQASM SDK	C++ /Python	Yes	Yes	AP
QNE-ADK	C++ /Python	Yes	No	AP

Hardware-oriented simulation tools allow the developer to model noise and physical entities with high accuracy. Main examples of hardware-oriented simulation tools are SQUANCH and NetSquid. Protocol-oriented simulation tools are described as focused on design and evaluation of general-purpose quantum protocols. As such, noise modeling is usually limited or not implemented. Protocol-oriented simulators are SeQUeNCe, QuISP and QuNetSim. Application-oriented simulation tools are made to ease the process of designing and implementing quantum network applications [11]. This paper also highlights the need for ideal quantum network or distributed quantum computing simulation tools.

Research [12] describes quantum networks in great detail, it also contains analysis of the quantum network simulation platforms which were available at the time of writing. Simulation platforms introduced in this paper are: SimulaQron, QuNetSim, SQUANCH, QuISP, SeQUeNCe and NetSquid.

QuNetSim and SimulaQron are described as geared towards network application development. QuNetSim is focused on ease of implementation of network applications. QuNetSim has simplified the process of synchronization and the logic at the node because it simulates the joint arrival of the quantum payload together with a corresponding classical header. QuNetSim was used to explore routing schemes in networks. SimulaQron is a simulation platform that can be run physically distinct machines to create a quantum computer network. SimulaQron was used to investigate entanglement verification schemes and various quantum protocols. SQUANCH is a platform designed to simulate quantum information protocols in large networks with realistic noise models. SQUANCH is based on agents, which are connected to other agents using quantum and classical channels, this type of agent based modeling enables parallelization [12].

Authors describe three discrete event simulators: QuISP, SeQUeNCe and NetSquid. QuISP. QuISP is a module developed on OMNeT++ classical network simulator. QuISP is described as a simulator used to better understand emergent behavior in large, interconnected quantum networks. SeQUeNCe is designed to simulate physical layer accurately and is highly modular. SeQUeNCe was used to simulate BB84 QKD protocol, Cascade protocol, quantum teleportation, and quantum network with nine nodes. NetSquid is also designed to enable physically accurate modeling. Nodes in NetSquid are modeled on NV centers in diamond. This simulation tool is also highly modular allowing developers to combine and change already existing components. NetSquid was used to evaluate a quantum router architecture [12]. The authors of this paper also highlight the need for continuous development of quantum network simulation tools, and their importance in scenarios of complex quantum networks.

3 The Role of QKD Network in Secure Communication

3.1 Distinguishing Characteristics of QKD and Classics Key Distribution

Quantum communication networks are built upon the principles of quantum mechanics to facilitate the transmission and manipulation of information. The foundational concepts and technologies employed in quantum communication networks encompass QKD, quantum teleportation, and quantum repeaters. QKD is a cryptographic key distribution technique that ensures secure transmission through the utilization of quantum mechanical principles. This approach relies on the inherent property that measuring a quantum system inevitably disrupts its state, thereby rendering any attempted interception of the key by eavesdroppers detectable. These concepts and technologies are currently undergoing active development and research aimed at enhancing their capabilities and exploring novel implementations in practical scenarios. The successful realization of these technologies is of utmost importance for the establishment of secure, high-capacity, and high-speed quantum communication networks.

Key distribution is a fundamental challenge in the field of cryptography, which involves securely sharing encryption keys between two parties. While a straightforward approach is to physically meet in a secure environment and exchange keys, modern cryptographic protocols utilize techniques such as public key cryptography, such as ciphers, RSA, Diffie-Hellman, to enable remote key exchange. However, conventional key distribution methods based on simple mathematical calculations are vulnerable to attacks by third parties. As such, developing secure and efficient key distribution protocols is a critical research area in cryptography [13].

3.2 QKD Network Parameters and Architecture

QKD enables two parties to produce and share a key that is then used for encrypting and decrypting messages. Specifically, QKD is a key distribution method that relies on the principles of quantum mechanics, and the working principal of QKD between two nods can be seen in Fig. 1 [14].

Currently, quantum network security does not appear as an independent application that provides complete protocols for secure communication. However, quantum key distribution techniques complement well-established internet technology. They are used in conjunction with the public internet or, more likely, with private networks that use a set of internet protocols to build secure communication systems. Such private networks are currently widely used worldwide by users who require secure and private communication, e.g. financial institutions, government organizations, militaries, etc., and that the integration of QKD technology with these types of private networks may prove feasible and immediately attractive in certain contexts [15]. Today, secure communication between cryptographic endpoints or even between individual computers on the Internet is enabled by a well-defined IPsec architecture. It defines the protocols, algorithms, databases, and policies necessary for secure communication. Therefore, it would be optimal to integrate QKD technology with the currently well-established internet security architecture. This joint effort would guarantee secure internet traffic via quantum cryptography.

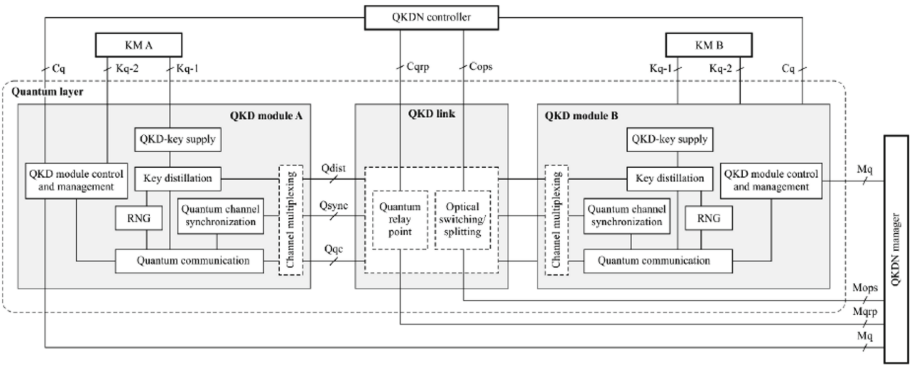

Fig. 1. The architecture of a system for QKD communication between two nodes [14]

Figure 2 elaborates one of possible scenarios of realizing QKD network as an upgrade to existing communication network infrastructure.

Quantum key distribution network consists of several key elements and characteristics that can become part of the simulation model. Fundamental element of a QKD network is a QKD link, representing logical connections between QKD nodes.Nodes are connected through quantum channel for entangled photons transmission and classical channel for synchronization and data exchange [15].

QKD network is represented by set of QKD links and QKD nodes and can be distinguished three layers, quantum layer, key management layer and communication layer [16]. QKD network has several specific requirements but also, as need to be integrated in existing environment it should meet several additional criteria. Link length is the criteria that defines the generation and transmission of secure key material. In current implementation with optical fibers (e.g. ITU-T G.652) the limitation distance for QKD links implementation is considered to be approximately 100 km [5, 17].

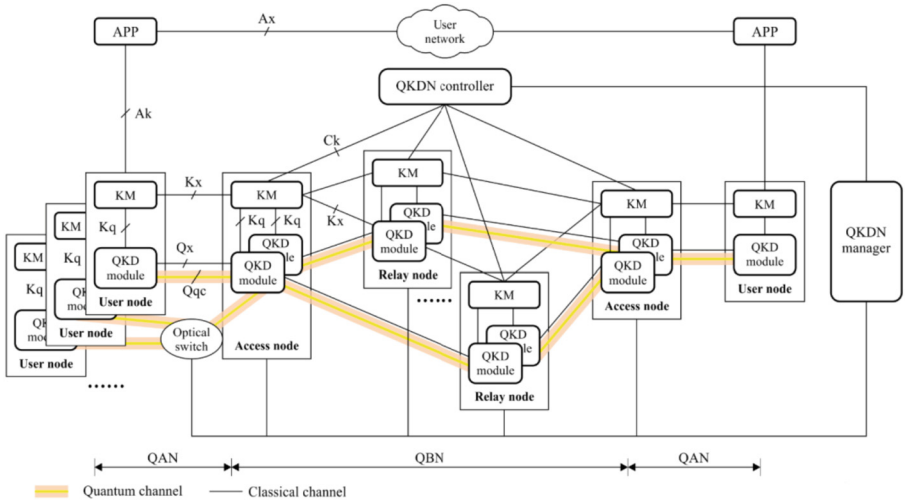

Fig. 2. QKD network implementation scenario [14]

The average key rate of a QKD link is a vital parameter that significantly impacts the performance of a QKD network as it determines the availability of key material for encryption and decryption operations. The competition between the rate of storing key material and its consumption directly affects network performance. Due to limited resources, communication within the network is minimized to conserve the previously established key material. To ensure security, communication typically occurs on the shortest routing path, reducing the number of nodes vulnerable to eavesdropping or abduction. Longer paths require more key material consumption. In cases of network congestion or communication issues, used key material is deliberately discarded, and new key material is applied for retransmission to mitigate the risk of leaks. Therefore, minimizing the number of hops is preferable to optimize network security [16].

The paramount interest in QKD arises from the need to ensure the privacy and uniqueness of the established key material, necessitating robust security measures at every level of the network architecture. This includes securing key material during its establishment, management, storage, and usage.

4 Overview of QKD Network Simulation Platforms

Based on the previous research and newly collected information, Table 3 contains currently available quantum network simulation platforms. Not all quantum network simulation platforms have the ability to simulate quantum key distribution, for that reason Table 3 contains implemented QKD protocols column which will be used to identify simulation platforms that enable QKD simulation.

Table 3. Overview of quantum network simulation platforms

Simulation tool	Language	Implemented QKD Protocols	Open Source	Discrete event simulation
SeQUeNCe	Python	BB84, Cascade	Yes	Yes
NetSquid	Python	All generic protocols	No	Yes
QuISP	C++	No	Yes	Yes
QuNetSim	Python	All generic protocols	Yes	No
SimulaQron	Python	N/A	Yes	No
SimQN	Python	BB84	Yes	Yes
ReQuSim	Python	N/A	Yes	Yes
SQUANCH	Python	No	Yes	No
QKDNetSim	C++	QKD Post-processing app	Yes	Yes
QNE-ADK	Python	Yes	No	No

4.1 Discrete-Event Simulation Platforms

Discrete-event simulation is a simulation paradigm that is used to model systems as a sequence of discrete events. It is based on events that define a change of state in the system – every individual event occurs at a specific time instance and records a change of state in the system. Discrete-event simulation allows for more accurate noise modelling since it depends on elapsed time. We give an overview of the following discrete-event simulation platforms.

SeQUeNCe is a scalable, modular, customizable, discrete-event quantum network simulator. It's open-source software that is available on GitHub [7]. SeQUeNCe has a modularized design that consists of six modules. First module is simulation kernel which enables discrete-event simulation. Second, hardware module consists of models of hardware components that are present in quantum networks, such as models of single-atom memories, light sources, single photon detectors, BSM nodes, quantum gates, quantum and classical channels. Third, entanglement management module enables high-fidelity end-to-end entanglement between network nodes. This module contains protocols for entanglement generation, purification and swapping. Fourth, resource management module is used to control local resources within one node. Resources are managed based on the commands issued by the network management module. Fifth, network management module provides quantum network services based on the requests from other modules. Sixth, the application module is designed to represent quantum network application and requests for resources. Examples of applications are quantum teleportation and quantum key distribution [7].

NetSquid is a simulation tool, developed at QuTech research center, designed for modelling and simulation of scalable quantum networks. NetSquid is freely available after registration, but it is not open-sourced. NetSquid is a software tool available as a

Python package, it is used to accurately simulate quantum networks and effects of physical non-idealities on them, such as networks where quantum entanglement is generated and utilized between nodes. Authors of this simulator implemented a discrete-event simulation engine – used to model delays during transmission and computation, specialized quantum computing library, framework for modelling of quantum hardware devices, and a framework for designing quantum protocols [6]. NetSquid is highly modular and as such it enables the use of already implemented hardware components as well as modification of said components.

QuISP is a discrete-event simulator that is designed to simulate and investigate behaviors of complex, large scale quantum networks with realistic noise models. QuISP is a module developed for OMNeT++ classical network simulator. Long term goal of this simulator is successful simulation of hundred networks, each containing hundred nodes. This simulation module was designed according to the following principles: realism – simulation needs to provide accurate information about the physical states that are distributed in the quantum network, scalability – complex, large scale quantum networks require high number of qubits, nodes and links between the nodes, flexibility – defined as high level of customizability which in turn ensures that QuISP will be able to accurately simulate present and future quantum hardware [18].

SimQN is a modular discrete-event network-layer simulation framework, packaged as Python library. This simulation framework is designed to facilitate large scale investigations of quantum networks. These simulations include QKD and entanglement distribution protocols, routing algorithms and resource allocation schemas. Authors of this simulation framework describe SimQN as general purpose, meaning it can be used to investigate QKD networks, entanglement distribution networks as well as other kinds of quantum networks [19]. As previously mentioned, goal of SimQN is to provide convenient way of simulating large scale quantum networks while balancing performance and functionality, this is achieved through modular design. Architecture of SimQN consists of several modules: a physical backend module, network utility module, network application module, quantum entities (nodes, quantum channel and classical channels) and other auxiliary modules [20].

ReQuSim is a simulation tool specifically designed to simulate quantum repeater networks. It is used to evaluate quantum repeater schemes to extend the distance of quantum key distribution and entanglement distribution protocols. ReQuSim enables entanglement purification, modeling of channel noise, multiple repeater links. ReQuSim is designed to combine realism and scale. ReQuSim can be used to model setups for quantum repeater, obtaining key rates for repeater protocols, evaluating effects of parameters on the performance of a repeater setup. This simulation tool is not designed to develop code that will be able to interface with future quantum hardware or simulate large scale networks containing thousands of nodes [21].

4.2 Non Discrete-Event Simulation Platforms

QuNetSim is a real time simulator written in Python and designed to ease the process of quantum network application development. Main goal of QuNetSim is to enable the users to quickly develop quantum networking protocols without having to invest time in

implementation of low-level software related tasks, these protocols include QKD, quantum money, anonymous transmission. This simulator uses a layered architecture inspired by the OSI model, these layers include application, transport, and network. QuNetSim includes network nodes that can be linked using both quantum and classical connection. QuNetSim allows users to create a network configuration of nodes connected via classical or quantum links and then program the behavior of each node in the network as they want [8]. Current implementation of QuNetSim still uses simplistic channel models which are not good for large quantum network simulations, smaller scale simulations containing less than ten nodes [8].

SimulaQron is a simulator used to develop distributed software that can run on simulated or real classical and quantum nodes. Goal of this tool is ease of development of network applications and exploration of software engineering practices in the context of quantum Internet. SimulaQron is not developed to achieve efficient simulation of large scale quantum networks and effects of noise, error correcting codes and similar models on said network [22].

SQUANCH is an open-source Python framework designed to create parallelized simulations of distributed quantum information processing. SQUANCH contains features of a general purpose quantum computing simulator, but it is specifically optimized to simulate quantum networks. SQUANCH can simulate realistic noise models over large networks. This simulator allows users to design complex multi-party quantum networks and create classes for modeling of noisy quantum channels. SQUANCH is based on agents which can manipulate a subset of a distributed quantum state, simulations are parallelized and every node is running local processes mirroring the distributed structure of a quantum network [9].

QNE-ADK is an application development kit that allows the users to create applications and experiments and run them on a simulator. QNE-ADK offers the users a command line interface with implemented commands that are used to create necessary files that define applications or experiments, these files are templates that are used to develop custom applications. This development kit also offers the users to use already existing applications and experiments which are accessible by using the command line interface and clone command. When creating custom application or experiment, users can set custom parameters, different node roles, as well as which channels and nodes will be used in the simulation. Once the experiment is configured and ready to run, the experiment is parsed and sent to the NetSquid simulator [23].

5 Analysis of QKD Network Simulators Characteristics

Based on the overview of currently available quantum network simulation platforms and tools, following platforms and tools are chosen for further analysis of their capabilities in the context of quantum key distribution network simulation: SeQUeNCe, NetSquid, QuNetSim, SimQN, ReQuSim, QNE-ADK.

Modularity is one of the most important properties for a QKD simulator. It enables researchers to develop and model their own protocols and networks in the same way one constructs a network in the real world. If the simulation toolkit is modular, it becomes easier to add new modules or interchange the existing ones, making the toolkit viable in the future as well.

Table 4. Characteristics and functionalities of QKD network simulation platforms

Simulation tool	Modular	Error models	Size of simulated network	Node types	Programming interface	Layer
SeQUeNCe	Yes	Yes	Small (<15 nodes)	Multiple	Python	Physical
NetSquid	Yes	Yes	Very Large (>1000 nodes)	General	Python	Physical
QuNetSim	No	No	Small (<15 nodes)	General	Python	Application
SimQN	Yes	Yes	Large (>400)	General	Python	Network
ReQuSim	Yes	Yes	Medium (N/A)	Station	Python	Network
QKDNetSim	Yes	N/A	N/A	QKD node	C++	Network
QNE-ADK	No	Yes	Small (<15 nodes)	General	Python	Application

Programming interface describes the programming language that is used to configure and run simulations in the corresponding simulation tool. Every simulation tool enables the users to use python as an interface, except QKDNetSim which uses C++. QKDNetSim also provides a graphical user interface through a web application that relays configuration settings to a simulator and displays simulation results.

Although quantum network simulators are developed with a focus on a specific layer, they also implement other layers to a varying degree of detail. Physical layer simulation tools are focused on detailed modeling of hardware components, while application layer tools leave the physical layer unspecified and focus on enabling the user to design and implement quantum network applications. Table 4 contains two application layer simulators – QuNetSim and QNE-ADK. Although the latter simulator is classified as an application layer, it communicates configuration settings to NetSquid that serves as a backend. QuNetSim also uses other simulators as a backend for physical simulation. User can configure which backend to use – SimulaQron [21], ProjectQ [24], or EQSN [25]. Network layer simulation tools are focused on enabling research focused on network management, entanglement distribution, routing in quantum networks, impact of different node configurations on the network, etc.

SeQUeNCe simulates hardware components with a high level of detail. It also provides the user with multiple types of network nodes. Network nodes can be general purpose, QKD node, quantum router, BSM node. QKD node contains protocol stack needed to create keys. SeQUeNCe implements BB84 and cascade QKD protocols – BB84 is used to generate a secure key between two nodes and cascade protocol is used to correct errors that occur while using BB84. Variety of nodes allows for high level of

customizability needed to create a detailed simulation. Modular design of SeQUeNCe provides the access to hardware components such as light source, memory, optical channel, photon, BSM node, etc. Some of the optical channel parameters that can be configured include length, attenuation and polarization fidelity. Topology that will be used in simulation can be generated through JSON file. To successfully create a topology JSON file needs to contain nodes – their names, node type and size of memory; quantum channels – connected nodes, attenuation value, distance and type; classical channels – connected nodes and delay value. At the time of writing, SeQUeNCe is equipped with graphical user interface designed to create simulations that is not fully operational, but it does show great potential. SeQUeNCe was used to simulate Chicago quantum network which was the largest topology containing nine nodes [7].

NetSquid also simulates hardware components and physical properties of quantum devices – quantum gates, quantum memory, to a great level of detail. Nodes in NetSquid are created through a general node class that can hold and manage hardware subcomponents. By combining this node class and creating custom or using already implemented protocols that are run on nodes, users can create different simulation experiments such as repeater chains [25]. Another NetSquid functionality that allows precise simulation of quantum network is delay modeling. Already implemented delay models are fixed delay model – fixed timing delay model, gaussian delay model – channel delay model with a Gaussian distribution and fiber delay model – transmission delay model based on constant speed of photons through fiber. NetSquid also allows to user to create custom quantum error models or use already implemented models. Implemented quantum error models are depolarization noise model, dephasing noise model, T1T2 noise model – phenomenological noise model based on T1 and T2 times, and fiber loss model – model used for exponential photon loss in optical fiber channels, it is based on length of channel. Largest NetSquid simulation contained a topology consisting of 1025 nodes forming a quantum repeater chain.

QNE-ADK is an application layer simulator. Customization in QNE-ADK is comprised of predetermined networks with included channels and nodes. There are three networks that users can choose from that contain five nodes and varying number of links between nodes (4, 5 and 10 links) creating different topologies. Users can change parameters such as gate fidelity on nodes, elementary link fidelity on network channels. These quantum network components and parameters are used by an application that the user can develop. Entanglement based QKD is one of the examples of already implemented applications in QNE-ADK where users can specify the number of EPR pairs that Alice and Bob will generate. Results of this simulation contain number of pairs Alice and Bob measured in the same basis, number of pairs chosen to compare measurement outcomes for, QBER – fraction of compared measurement outcomes that are not equal, although they result from measurement in the same basis and raw keys that Alice and Bob possess. These simulations and results are visualized and displayed through a web application called quantum network explorer that contains graphical user interface.

QuNetSim is another application layer simulator that is not designed to simulate realistic physical models of quantum hardware and channels. This simulation tool is focused on enabling the users to develop and test protocols for quantum networks. QuNetSim doesn't use discrete event simulation engine which is necessary to model

and track noise that is time dependent. Currently this simulation tool is used for smaller scale simulations consisting of five to ten nodes. Nodes in QuNetSim are called hosts and they can be configured to function as a quantum node, a relaying node or an eavesdropper.

SimQN is a network layer simulation tool that enables large scale simulations. One of the example use cases of SimQN is routing algorithm design for better QKD performance [26]. SimQN uses node class to describe quantum network nodes. Functionalities of these nodes can be changed by equipping them with ither devices like memories. SimQN contains already implemented error models such as dephasing and depolarization error models that can be applied to quantum memory, quantum channels. This simulation tool also allows access to delay models – normal, uniform, and constant delay models.

ReQuSim is another network layer simulation tool designed to investigate quantum repeater strategies. Nodes in ReQuSim are designed to be quantum repeater stations. Channels in this simulation tool can be equipped with noise models to simulate a network with higher level of detail.

QKDNetSim is a specialized simulation tool focused on simulating quantum key distribution, more specifically it is focused on key usage. QKDNetSim is designed with a simplified QKD link and focuses on network performance measurement and estimation, traffic management, key consumption, and routing protocols [27]. Simulation in QKDNetSim can contain unlimited number of nodes. QKDNetSim is a module developed in the NS-3 simulator. As it was previously mentioned QKDNetSim can be accessed through a web interface where users can set up a simulation. Simulations consist of setting up QKD nodes on a desired location and connecting them with QKD links. These links contain changeable parameters: key rate, key size, packet size, etc. After placing the nodes and connecting them, users can place an applications that consume secret keys between two nodes. Parameters of applications are authentication type, encryption type, application packet size, application traffic rate, etc. Furthermore, application interface with key management service is described by ETSI GS QKD 014 and ETSI GS QKD 004 specifications. Results of simulation contain information about relayed, consumed and generated key pairs for each QKD link. Results also contain statistics about consumed key pairs for each added application and statistics about exchanged packets between application and key management service, application to application signaling packets and application to application data packets.

6 Conclusion

In the research presented in this paper we studied and analyzed currently available tools and platforms applicable for simulation of QKD networks. Simulation plays a crucial role in the design and pre-implementation stages of QKD networks. By utilizing simulation tools and platforms, researchers and engineers can gain valuable insights into the behavior and performance of these networks without the need for costly and time-consuming physical implementations.

The findings of this research have shed light on the limitations present in all the analyzed tools, which resulted from insufficient research engagement in this field. It is important to note that despite the numerous challenges and identified shortcomings, it is possible to establish a simulation model and conduct simulations at certain levels

by combining the functionalities of multiple tools. However, it is evident that this area requires further research efforts and engagement in the development of existing tools, as well as the creation of new ones that would provide comprehensive functionality. The holistic approach is essential to facilitate the design and implementation of simulation models for QKD networks in the future.

References

1. Aji, A., Jain, K., Krishnan, P.: A survey of Quantum Key Distribution (QKD) network simulation platforms. In: 2021 2nd Global Conference for Advancement in Technology (GCAT), pp. 1–8. IEEE (2021). https://doi.org/10.1109/GCAT52182.2021.9587708
2. Zukarnain, Z.A., Buhari, A., Harun, N.Z., Khalid, R.: QuCCs: an experimental of quantum key distribution using quantum cryptography and communication simulator. In: The 6th International Cryptology and Information Security Conference, pp. 127–40 (2019)
3. Chatterjee, R., Joarder, K., Chatterjee, S., Sanders, Barry C., Sinha, U.: QkdSim, a simulation toolkit for Quantum Key Distribution including imperfections: performance analysis and demonstration of the B92 protocol using heralded photons. Phys. Rev. Appl. **14**(2) (2020). https://doi.org/10.1103/PhysRevApplied.14.024036
4. Abdelgawad, M.S., Shenouda, B.A., Abdullatif, S.O.: EnQuad: a publicly-available simulator for Quantum Key Distribution protocols. Cybern. Inform. Technol. **20**, 21–35 (2020). https://doi.org/10.2478/cait-2020-0002
5. Fan-Yuan, G.-J., Chen, W., Lu, F.-Y., Yin, Z.-Q., Wang, S., Guo, G.-C., et al.: A universal simulating framework for Quantum Key Distribution systems. Science China Inf. Sci. **63**, 180504 (2020). https://doi.org/10.1007/s11432-020-2886-x
6. Kreinberg, S., Koltchanov, I., Novik, P., Alreesh, S., Laudenbach, F., Pacher, C., et al.: Modelling weak-coherent CV-QKD systems using a classical simulation framework. In: 2019 21st International Conference on Transparent Optical Networks (ICTON), pp. 1–4. IEEE (2019). https://doi.org/10.1109/ICTON.2019.8840253
7. Wu, X., Kolar, A., Chung, J., Jin, D., Zhong, T., Kettimuthu, R., et al.: SeQUeNCe: a customizable discrete-event simulator of quantum networks. Quant. Sci. Technol. **6**, 045027 (2021). https://doi.org/10.1088/2058-9565/ac22f6
8. DiAdamo, S., Nötzel, J., Zanger, B., Beşe, M.M.: QuNetSim: A Software Framework for Quantum Networks (2020). https://doi.org/10.1109/TQE.2021.3092395
9. Bartlett, B.: A distributed simulation framework for quantum networks and channels (2018)
10. Wu, X., Zhang, B., Jin, D.: Parallel simulation of Quantum Key Distribution networks. In: Proceedings of the 2020 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, pp. 187–196. ACM, New York, NY, USA (2020). https://doi.org/10.1145/3384441.3395988
11. Caleffi, M., Amoretti, M., Ferrari, D., Cuomo, D., Illiano, J., Manzalini, A., et al.: Distributed Quantum Computing: a survey (2022)
12. Azuma, K., Bäuml, S., Coopmans, T., Elkouss, D., Li, B.: Tools for quantum network design. AVS Quant. Sci. **3**(1) (2021). https://doi.org/10.1116/5.0024062
13. Kozlowski, W., Dahlberg, A., Wehner, S.: Designing a quantum network protocol. CoNEXT 2020 - Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies, pp. 1–16. Association for Computing Machinery, Inc (2020). https://doi.org/10.1145/3386367.3431293
14. ITU-T. Quantum key distribution networks – functional architecture (2020)
15. Mehic, M., Niemiec, M., Rass, S., Ma, J., Peev, M., Aguado, A., et al.: Quantum Key Distribution: a networking perspective. ACM Comput. Surv. **53** (2020). https://doi.org/10.1145/3402192

16. Tysowski, P.K., Ling, X., Lütkenhaus, N., Mosca, M.: The engineering of a scalable multi-site communications system utilizing Quantum Key Distribution (QKD). Quant. Sci. Technol. (2017). https://doi.org/10.1088/2058-9565/aa9a5d
17. Ribezzo, D., Zahidy, M., Vagniluca, I., Biagi, N., Francesconi, S., Occhipinti, T., et al.: Deploying an inter-European quantum network. Adv. Quant. Technol. **6**(2), 2200061 (2022). https://doi.org/10.1002/qute.202200061
18. Satoh, R., Hajdušek, M., Benchasattabuse, N., Nagayama, S., Teramoto, K., Matsuo, T., et al.: QuISP: a Quantum Internet Simulation Package (2021). https://doi.org/10.1109/QCE53715.2022.00056
19. Chen, L., Li, J., Xue, K., Yu, N., Li, R.: A discrete time scheduler designed for Quantum Network n.d. https://github.com/ertuil/SimQN. Accessed 30 May 2023
20. Chen, L., Xue, K., Li, J., Yu, N., Li, R., Sun, Q., et al.: SimQN: a network-layer simulator for the quantum network investigation. IEEE Netw. 1–8 (2023). https://doi.org/10.1109/MNET.130.2200481
21. Wallnöfer, J., Hahn, F., Wiesner, F., Walk, N., Eisert, J.: ReQuSim: faithfully simulating near-term quantum repeaters (2022)
22. Dahlberg, A., Wehner, S.: SimulaQron - a simulator for developing quantum internet software (2017). https://doi.org/10.1088/2058-9565/aad56e
23. Application Development Kit for Quantum Network Explorer n.d. https://github.com/QuTech-Delft/qne-adk. Accessed 30 May 2023
24. Steiger, D.S., Häner, T., Troyer, M.: ProjectQ: an open source software framework for quantum computing. Quantum **2**, 49 (2018). https://doi.org/10.22331/q-2018-01-31-49
25. Coopmans, T., et al.: Simulation of a 1025-node quantum repeater chain of NV centres with NetSquid, a new discrete-event quantum-network simulator (2019). https://ui.adsabs.harvard.edu/abs/2019APS..MARL28012C/abstract. Accessed 31 May 2023
26. Elliot Chen. SimQN (2022). https://ertuil.github.io/SimQN/introduction.html. Accessed 30 April 2023
27. Mehic, M., Maurhart, O., Rass, S., Voznak, M.: Implementation of quantum key distribution network simulation module in the network simulator NS-3. Quant. Inf. Process. **16**, 253 (2017). https://doi.org/10.1007/s11128-017-1702-z