Quantum Inf Process (2017) 16:253
DOI 10.1007/s11128-017-1702-z

Implementation of quantum key distribution network simulation module in the network simulator NS-3

Miralem Mehic[1] · **Oliver Maurhart**[2] · **Stefan Rass**[3] · **Miroslav Voznak**[1]

Received: 27 January 2017 / Accepted: 18 August 2017 / Published online: 30 August 2017

Abstract As the research in quantum key distribution (QKD) technology grows larger and becomes more complex, the need for highly accurate and scalable simulation technologies becomes important to assess the practical feasibility and foresee difficulties in the practical implementation of theoretical achievements. Due to the specificity of the QKD link which requires optical and Internet connection between the network nodes, to deploy a complete testbed containing multiple network hosts and links to validate and verify a certain network algorithm or protocol would be very costly. Network simulators in these circumstances save vast amounts of money and time in accomplishing such a task. The simulation environment offers the creation of complex network topologies, a high degree of control and repeatable experiments, which in turn allows researchers to conduct experiments and confirm their results. In this paper, we described the design of the QKD network simulation module which was developed in the network simulator of version 3 (NS-3). The module supports simulation of the QKD network in an overlay mode or in a single TCP/IP mode. Therefore, it can be used to simulate other network technologies regardless of QKD.

Keywords Quantum key distribution · Network · Simulation · NS-3

✉ Miralem Mehic
miralem.mehic.st@vsb.cz

[1] Department of Telecommunications, VSB-Technical University of Ostrava, 17. listopadu 15, 708 00 Ostrava-Poruba, Czech Republic

[2] Digital Safety and Security Department, AIT Austrian Institute of Technology GmbH, Donau-City-Strasse 1, 1220 Vienna, Austria

[3] Universitaet Klagenfurt, Institute of Applied Informatics, System Security Group, Universitaetsstr. 65-67, 9020 Klagenfurt, Austria

1 Introduction

After the design of a new network solution, a researcher has typically several possibilities to evaluate and validate the obtained results. Analytically quantifying the performance and complex behaviour of even simple network protocols in the aggregate is often impractical. It uses mathematical models to evaluate network performance where queuing theory is one of the most common tools in network performance studies. Unfortunately, the theoretical analysis of networks containing a large number of nodes and links is a demanding process, since the mathematical constructs become too complex for realistic considerations. A simulation is an essential tool for the computer networks research. The simulation environment offers the creation of complex network topologies, a high degree of control and repeatable experiments, which in turn allows researchers to conduct experiments and confirm their results. Also, the simulator provides an easy way to manipulate desired parameters while setting other parameters fixed which provides a simpler and more comfortable way of looking at the problem, with a further reduction of costs in relation to the practical testbed.

Quantum key distribution (QKD), based on the laws of physics rather than the computational complexity of mathematical problems, provides an information-theoretically secure (ITS) way of establishing symmetrical binary keys between two geographically distant users [1–3]. The keys are secure from eavesdropping during the transmission and QKD ensures that any third party's knowledge of the key is reduced to a minimum [4,5]. In recent years, noticeable progress in the development of quantum equipment has been reflected through a number of successful demonstrations of the QKD technology [6–11]. While they show the great achievements of QKD, many practical difficulties still need to be resolved. Taking into account the high cost of deploying a QKD network testbed, we present here the first computing simulation environment of the QKD network.

This article follows up on our previous work [12] and it has been written to provide an insight into the practical realization of the QKD network, that is, integration of QKD with the existing network technologies. In this paper, we describe the design of a simulation module of the QKD network which was implemented in the NS-3 simulator of version 3.26 [13]. Our module allows simulating the QKD network with multiple nodes and links using various network applications. This article is organized as follows: Sect. 2 outlines the fundamentals of QKD including the state of the art. Section 3 describes the QKD network simulation module and provides a simple example of use, while Sect. 4 concludes this study and outlines the future work.

2 Fundamentals of quantum key distribution

QKD networks differ from the traditional communication network in several aspects. One of the main differences is reflected in the implementation of the network link. A QKD link employs two distinct communication channels between the parties: the quantum channel, which is used for transmission of quantum key material encoded in certain photon properties such as polarization or phase, and the public channel, which is used for verification of the exchanged key material and transmission of the encrypted

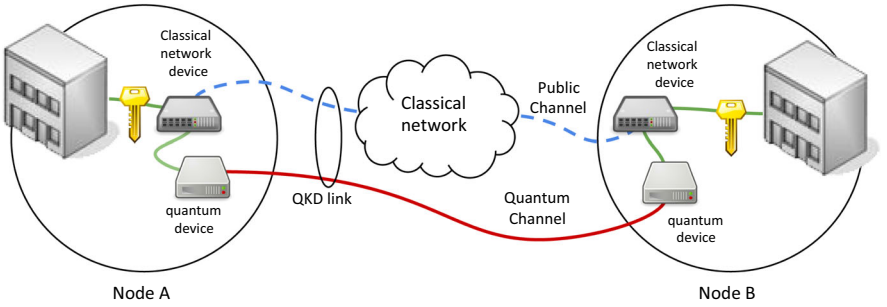

Fig. 1 Overview of a QKD link between two QKD nodes which consist of an optical/quantum channel (*continuous red line*) and a public/classical channel (*dashed blue line*) (Color figure online)

data (Fig. 1). A quantum channel is always a point-to-point connection between exactly two nodes [14] while a public channel can be realized as any conventional connection which may include an arbitrary number of intermediate devices [15,16]. In practice, the quantum and the public channel can be implemented on the same media using existing optical components [17,18].

A second important feature of the QKD network is reflected in the limited length of the links and the maximal transfer rate. Namely due to absorption and scattering of polarized photons [19–22], the quantum channel can only be realized for a certain distance. An equally important feature of the link is the amount of key material that can be established in a unit of time and this amount may vary due to humidity, temperature, the stability of devices, global radiation, pressure, dust, sunshine duration or other factors [14,23]. However, the key rate mostly depends on the length of the link and it is often referred to as the key generation rate or simply key rate. Although key rate results of up to 1 Mbps have been achieved [24–27], such solutions are limited to very short distances. For current systems, the distance at which a QKD link is possible is roughly limited to 100 km in optical fibres, while the stable key rate is currently restricted to a few tens or hundreds of kbps depending on the distance [21,23].

Due to the limited key rate, the links are organized in the following way: both endpoints of the corresponding link have key storages (buffers) with limited capacity which are gradually filled with the new key material, which is subsequently used for the encryption/decryption of the data flow. The QKD devices constantly generate keys at their maximum key rate until the key storages are filled [28]. The type of the used encryption algorithm and the amount of network traffic to be encrypted determines the speed of emptying the key storage, often referred as the key consumption rate, while the key rate of the link determines the key charging rate [7,14,29]. If there is no enough of the stored key material, the encryption of the data flow cannot be performed [30] and the link can be characterized as "currently unavailable" To provide an ITS communication, the key tends to be applied with a One-Time Pad (OTP) cipher and authenticated using an ITS message authentication scheme such as Wegman–Carter when communicating over the public channel [31,32]. As a result, the ITS communication requires more bits of key material than the length of the secured message [33,34]. If the ratio between the charging and the consumption rates is not appropriate OTP cannot be used due to the

lack of the key material and using less secure algorithms that do not require too much material such as Advanced Encryption Standard (AES) becomes inevitable [35].

QKD networks also differ from conventional networks in terms of the network organization. Although there are theoretical and pioneering results in the field of quantum repeaters and quantum relays [36–38], in practice they remain unachievable with the current technology [2,21]. Therefore, the communication within the network usually takes place in a hop-by-hop [39] or in a key relay manner [7,40], which mainly differ in the way in which the connection is established. To put it simply, the packet is encrypted on one side of the link using a symmetrical encryption key and decrypted on the other side of the corresponding link using the same encryption key. This procedure is repeated for each node on the path until the transported message reaches its final destination. Both methods rely on the assumption that all nodes along the path between the sender and the receiver are fully trusted [30,41]. However, this restriction can be overcome when multiple pathbased communication or Quantum Network Coding [42,43] is used.

To facilitate the practical implementation of the requirement to bypass nontrusted nodes, in practice, the QKD network is usually deployed as an overlay point-to-point network [14,28,44]. An overlay network utilizes the services of an existing underlying network in an attempt to implement better services and one of its most important features is the independence of the path that is offered by the Internet service provider (ISP). The crux of this technology is to find an alternative route enabling to avoid the poor-quality routes, quickly switch communication via this route or even use multiple paths at the same time. Interconnected autonomous systems usually exchange routing information using an exterior routing protocol such as Border Gateway Protocol (BGP) [45], which is known to be slow in reacting and recovering from network events. Previous measurement studies have shown that BGP may take tens of minutes to reach a consistent view of the network topology after a failure [46]. In addition, since BGP advertises only one route, network nodes are prevented from seeing alternative paths, including the paths they might prefer. Due to the possibility to circumvent such problems, overlay networks are typically spawned between end-nodes that share resources with each other in a peer-to-peer (P2P) fashion which means that the network traffic is encapsulated into the traffic of the underlying network. The overlay networks behave selfishly in order to optimize its performance. This, however, can cause a reaction of the underlying network that seeks to favour some other network services [47,48]. The struggle for network resources can lead to dynamic and unpredictable QKD link performances [49].

Unlike conventional networks, there are few software applications dealing with QKD. The Quantum Cryptography Protocol Simulator [50] developed using C/C++ architecture is able to analyse the quantum bit error rate (QBER) and eavesdropper influence on the performances of the quantum channel when BB84 or B92 QKD protocol is used. A similar application is reported in [51]. An object-oriented simulation for QKD protocols was reported in [52] while an event-by-event simulation model and polarizer as a simulated component for QKD protocols with the presence of eavesdropper and misalignment measurement as scenarios was reported in [53]. A simulation framework for the QKD protocols using OptiSystem was reported in [54], and a modelling framework designed to support the development and performance

analysis of practically oriented QKD system representations was reported in [55]. To the best of our knowledge, the applications for simulating the QKD networks with multiple nodes and links are not available.

3 The QKD network simulation module

In contrast to the previously developed simulation tools focused on the quantum channel and QKD protocols, the QKD Network Simulation Module (QKDNetSim) described in this paper focuses on the public channel and simulation of QKD network. Hereof, QKDNetSim seeks for a maximum simplification of QKD link and puts the focus on network performance measurement and estimation, routing protocols, packet encapsulation and traffic management including generation and consumption of key material. More specifically, QKDNetSim is intended to facilitate additional understanding of QKD technology with respect to the existing network solutions. It seeks to serve as the natural playground for taking the further steps into this research direction (even towards the practical exploitation in subsequent projects or product design).

3.1 Requirements

The aim of the QKDNetSim project was not to develop the entire simulator from scratch but to develop the QKD simulation module in some of the already existing well-proven simulators. Therefore, QKDNetSim was developed to meet the following fundamental requirements:

3.1.1 Accuracy

Although the primary goal of the simulation is to use abstraction to reduce the complexity of the analysed problem, a very important aspect is the accuracy and the credibility of the obtained results. A good simulator is credible if it is validated and reliable network design decisions can be based on it [56–58]. It is required from the simulator to reflect the real-world scenarios as much as possible. However, due to the high degree of abstraction of the reality which carries a significant simplification as well as faster simulating performances, simulators are often unable to model the real problems and they do not fit precisely with real-world measurements. To reduce the level of abstraction and to bring the obtained results closer to the real-world measurements, while retaining the benefits of the simulated environment, the concept of the emulators has been introduced. An important advantage of the emulation environments over the simulation environments is the possibility to validate against the real traffic [59]. On the other hand, the advantage of the emulation environments over the real-world experiments is the possibility to scale to larger topologies by multiplexing the simulated elements on physical resources such as network interfaces and other.

Therefore, as a basic simulation platform, we sought one that is well- tested and well-proven. We looked for a credible simulator that is already well accepted in the scientific and research community, allows reliable simulations and has the emulation capacity to enable using the developed QKD model in practical testbeds.

3.1.2 Extensibility

Expandability is reflected in terms of the ease of upgrade, that is, implementation of the new models and solutions in the field of QKD and other technologies [31,60–63]. Without this, the development of the solution that is neither modular, nor does it provide a useful baseline for the upgrade will quickly lead to uselessness. From a software engineering perspective, this implies the development of a common reusable code which can be tailored to various needs [64].

3.1.3 Usability

Usability is reflected in the aspect of the ease of use and integration with the existing solutions [59,64]. A very important aspect is the software support, that is, the active cooperation of developers which entails discussions and latest updates, error reporting, clear and concise instructions and user manuals.

3.1.4 Availability

The QKD Network simulation module described in this paper will be distributed as open-source freeware and should leverage and permit the inclusion of other free and open-source networking software. Therefore, we looked for a simulation platform that is open-source and available for extensions. The aim was to provide an opportunity for the integration and testing of the QKD technology with a variety of existing network solutions by minimizing the cost of the simulation as much as possible.

3.2 The network simulator NS-3

Network Simulator of version 3 (NS-3) is an open-source software which is licensed under GNU GPLv2 and welcomes developers in the contributing code from across the academic world, industry, and government [13,59]. The development of the simulator is followed by actively encouraging the community participation by providing an open mailing list for user and active developer discussions, a tracker for the error reporting and a wiki with user-contributed instructions. Also, NS-3 comes with the instructions and descriptions of all simulator elements in the detailed *doxygen* web edition [65].

NS-3 is a discrete-event simulation written entirely in C++, with optional Python bindings, architected similar to Linux computers. NS-3 emphasizes the emulation capabilities that allow NS-3 to be used on testbeds and with real physical devices and applications. This is achieved by introducing the Emu NetDevice component which allows NS-3 simulations to send data on a physical network. In addition, a Tap NetDevice allows a host from the physical network to participate in a NS-3 simulation as if it were one of the simulated nodes. A NS-3 simulation may be constructed with any combination of simulated, Emu, or Tap devices. NS-3 consists of several already developed simulation modules such as Wifi, WiMAX, LTE, point-to-point, UAN and other [59,66], which can be used to reasonably handle matters of the public channel in QKD network. NS-3 has a widespread use reported in the scientific literature and it has

been proved to be a quality successor to the previously popular NS-2 simulator [59]. Taking this into account, we opted for NS-3 as the basic simulation platform.

3.3 QKDNetSim design

Due to the nature of the single photon propagation, QKD networks are mainly limited to the metropolitan scale [10,17,67] in which the network can be geographically divided into multiple domains (autonomous systems), or in the simplest case it can be a simple network in a single domain. Clearly, the geographic distribution of the network dictates the addressing and routing between nodes. Taking the assumption that these networks are not limited only to the QKD traffic (i.e. the network is utilized for various types of applications which can lead to dynamic and unpredictable link performance), the implementation of a routing protocol between multiple domains may not be a trivial task. In a simplified scenario, in which the network is implemented in a single domain and is exclusively intended for the QKD traffic, the problem of addressing and routing is much simplified. To support different simulation scenarios in various situations, our network simulation model allows the simulation of QKD network in both cases.

3.3.1 QKD key

The QKD key is an elementary class of QKDNetSim. It is used to describe the key that is established in the QKD process. The QKD key is characterized by several metakey parameters, of which the most important are the following:

- key identification (ID)
- key size
- key value in *std::string* or *byte* format
- key generation timestamp

3.3.2 QKD buffer

QKD keys are stored in QKD buffers which are characterized by the following parameters:

- QKD is also known as Quantum Key Growing [68,69] since it needs a small amount of key material preshared between the parties to establish a larger amount of the secret key material. The preshared secret key serves to guarantee the integrity of the protocol in the first transaction and it should not be used for any other purposes except to establish a new key material [14,29]. The amount of preshared key material for that purpose is denoted with $M_{\min}$,
- The key material storage depth $M_{\max}$, used to denote the maximal amount of keys that can be stored in QKD buffer,
- The current value $M_{\text{cur}}(t)$, representing the amount of key material in QKD buffer at the time of measurement t, where it holds that $M_{\text{cur}}(t) \leq M_{\max}$
- The threshold value $M_{\text{thr}}(t)$ at the time of measurement t is used to indicate the state of QKD buffer where it holds that $M_{\text{thr}}(t) \leq M_{\max}$. During the simulation,

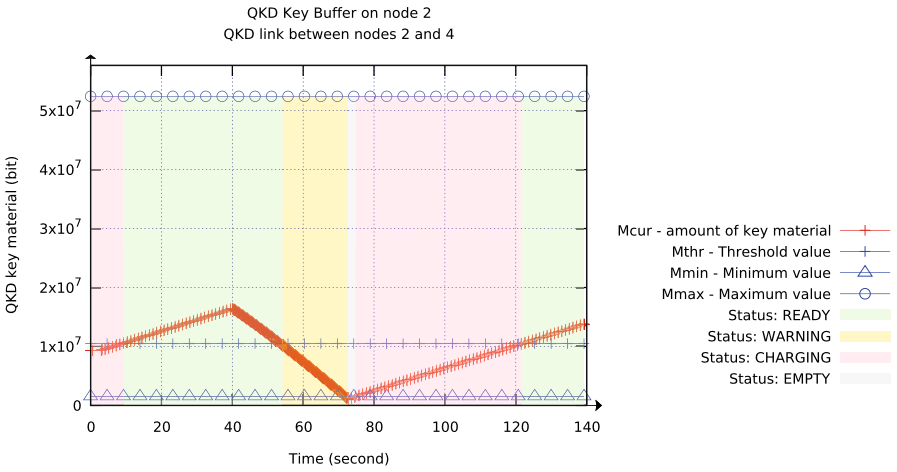

Fig. 2 Graphical representation of QKD buffer status generated using the QKD Graph

$M_{\text{thr}}(t)$ can be static or variable depending on the state of the network. However, algorithms for calculation of M_{thr} are out of the scope of this paper.

As shown in Fig. 2, the QKD buffer can be in one of the following states:

- READY—when $M_{\text{cur}}(t) \geq M_{\text{thr}}$,
- WARNING—when $M_{\text{thr}} > M_{\text{cur}}(t) > M_{\text{min}}$ and the previous state was READY,
- CHARGING—when $M_{\text{thr}} > M_{cur}(t)$ and the previous state was EMPTY,
- EMTPY—when $M_{\text{min}} \geq M_{\text{cur}}(t)$ and the previous state was WARNING or CHARGING.

By default, the state of QKD buffer does not directly affect the communication, but it can be used for easier prioritization of the traffic depending on the state of the buffer. For example, in EMPTY state, QKD post-processing application used to establish the new key material should have the highest priority in traffic processing. QKD post-processing applications are discussed in Sect. 3.3.6.

Without losing the generality, QKDNetSim allows to store the virtual key material in QKD buffers instead of the actual occupation of memory by establishing and storing the symmetrical key material in QKD buffers. To put it simply, the key material is not generated, nor does it take up computer memory, it is only represented by a number that indicates the amount of key material in the QKD buffer. In network simulators, such operations are common since they reduce the duration of the simulation and save computational resources. For example, instead of generating packets with random packet payload, network simulators often generate empty packets. However, QKDNetSim allows the user to turn off this level of abstraction in a way to choose the real key material generation and storage in QKD buffers.

3.3.3 QKD crypto

QKD crypto is a class used to perform encryption, decryption, authentication, authentication-check operations and reassembly of previously fragmented packets.

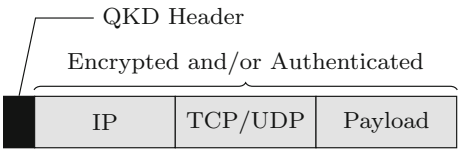

Fig. 3 QKD packet encapsulation

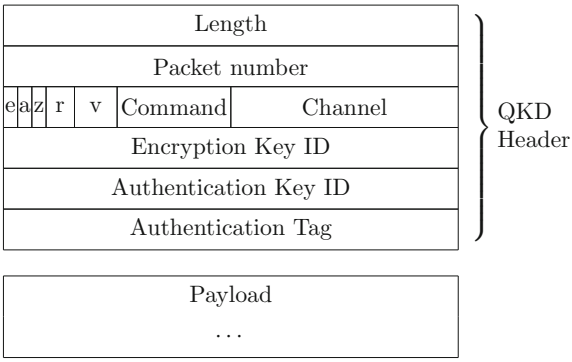

Fig. 4 QKD header

QKD crypto uses cryptographic algorithms and schemes from *Crypto++* open-source C++ class cryptographic library [70]. Currently, QKD crypto supports several cryptographic algorithms and cryptographic hashes including One-Time Pad (OTP) cipher, Advanced Encryption Standard (AES) block cipher, VMAC message authentication code (MAC) algorithm and others. Also, QKD crypto implements functions for serialization and deserialization of the packet into a byte array which is used as the input in cryptographic algorithms and schemes.

3.3.4 QKD virtual network device

To facilitate the ease of routing operation, in practice, encryption and authentication are performed at data link ISO/OSI layer. As shown in Fig. 3, the packet is encapsulated using a QKD header which contains authentication tag. Note that the QKD header is authenticated but it is not encrypted by default [14,71]. Due to the lack of the QKD header standardization, QKDNetSim implements the QKD Header which is implemented in AIT R10 QKD post-processing software [71] depicted in Fig. 4 while Table 1 provides a short explanation of QKD header's fields.

QKDNetSim allows realizing an overlay network which can be used for various purposes regardless of the QKD network. To ensure the independence of the underlying network, each QKD node implements an overlay TCP/IP stack with an independent overlay routing protocol as shown in Fig. 5. During the development of QKDNetSim, we aimed at minimizing the changes to the existing core code of the NS-3 simulator. However, to ensure independent overlay networking, QKDNetSim implements additional classes in the *internet* module of the NS-3 simulator. Class *ipv4-protocol* keeps a list of all IPv4 address associated with IPv4 interfaces. Thus, to distinguish IP addresses of underlying and overlying network, *virtual-ipv4-protocol* class is introduced. This

Table 1 QKD header fields

Field	Length	Description
Length	32 bits	Total packet length in bytes
Packet number	32 bits	The packet number
e	1 bit	Encrypted
a	1 bit	Authenticated
z	1 bits	Compressed (zipped)
r	2 bits	Reserved for further use
v	3 bits	Version
Command	4 bits	Control packet command
Channel	4 bits	The channel ID
Encryption key ID	32 bits	ID of encryption key
Authentication key ID	32 bits	ID of authentication key
Authentication tag	32 bits	Authentication tag
Payload	–	Data payload

implies the implementation of independent overlay TCP and UDP L4 protocol classes which pass packets to *virtual-ipv4-protocol* instead of original *ipv4-protocol*. Additionally, the realization of independent TCP and UDP L4 protocol classes allows for simple modifications of overlying L4 communication without affecting the underlying TCP/IP stack.

In an overlay network, QKD NetDevices are usually registered as Virtual QKD NetDevice since the MAC header is replaced with a QKD header as shown on Fig. 5. However, in case the QKD network is realized as a network with a single TCP/IP stack, the packet contains QKD and MAC headers. QKD NetDevice implements a sniffer trace source which allows recording of the overlay traffic in *pcap* trace files. Also, QKD NetDevice allows fine tuning of the MAC-level Maximum Transmission Unit (MTU) parameter which limits the size of the frame in the overlying network [62].

In QKDNetSim, the bond with QKD crypto is realized via the QKD manager. Packets leaving QKDNetDevice are passed to the QKD manager which is in charge of control of the cryptographic process. Similarly, the reception packet is processed (authentication check and/or decryption) before passing into the QKD NetDevice. Such implementation allows the use of other types of devices instead of QKD NetDevices, that is, the use of different network technologies such as Point-to-Point, WiFi, WiMAX, LTE, UAN and other. In the case of the network with a single TCP/IP stack, the QKD manager is called from Traffic Control Layer that sits between NetDevice (L2) and IP protocol (L3). Placing a connection to the QKD manager in the same layer with waiting queues of Traffic Control Layer allows for a simple usage of various types of NetDevices and it follows up on the previous work on the encryption of the packet content above the OSI data link layer (Fig. 5) [28,62,71].

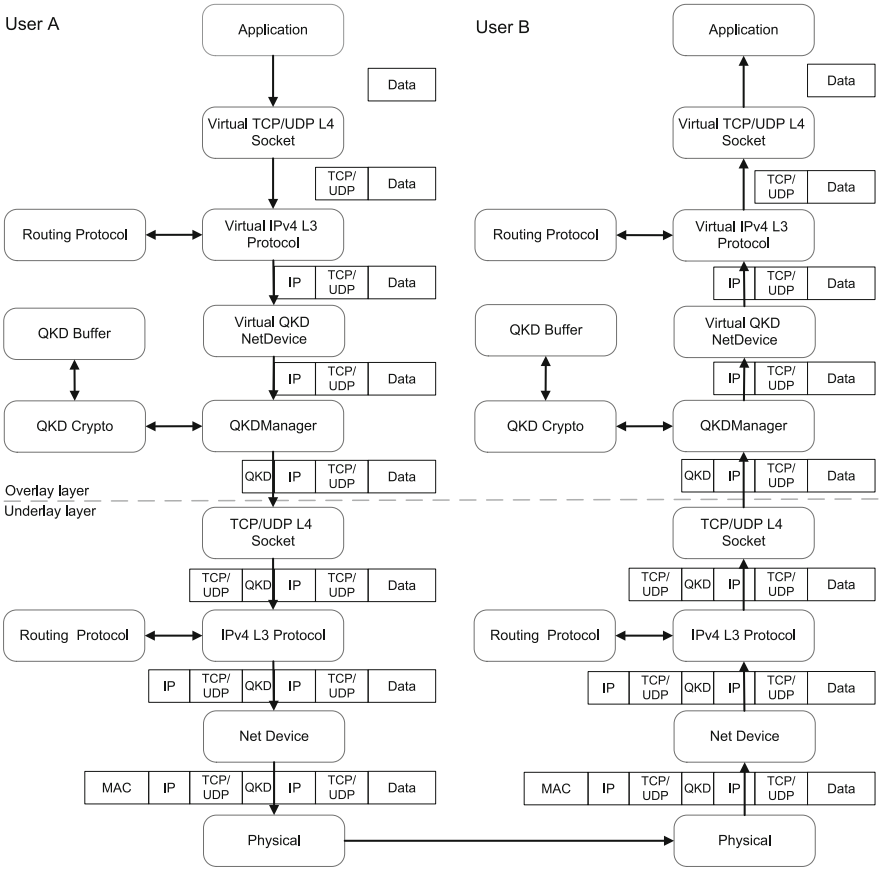

Fig. 5 Packet encapsulation in the QKD overlay network

3.3.5 QKD manager

The QKD manager is installed at each QKD node and it represents the backbone of QKDNetSim. It contains a list of QKD Virtual NetDevices from the overlying network, a list of active sockets in the underlying network, a list of IP addresses of interfaces in the overlying and underlying network and a list of associated QKD buffers and QKD cryptos. Therefore, the QKD manager serves as a bond between the overlying NetDevices and sockets in the underlying network. Since QKD link is always realized in a point-to-point manner between exactly two nodes [14], QKD manager stores information about NetDevices of the corresponding link and associated QKD buffers. Using the MAC address of NetDevice, QKD manager unambiguously distinguishes QKD crypto and QKD buffer which are utilized for packet processing. Finally, QKD manager delivers the processed packet to the underlying network. Receiving and processing the incoming packets follows the identical procedure but in a reverse order.

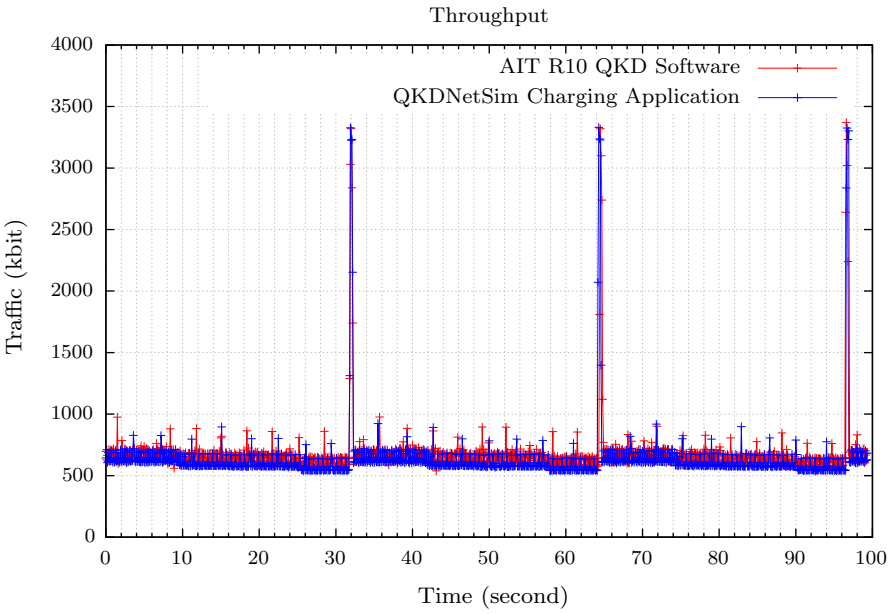

Fig. 6 Comparison of the traffic generated using QKDNetSim post-processing (charging) application and AIT R10 QKD software that was used for the post-processing of the keys in the AIT quantum laboratory in April 2016. The traffic shown here was recorded in a local network, that is, excluding the traffic generated by any other application

3.3.6 QKD post-processing application

Key material establishment process is an inevitable part of the QKD network. It is performed using QKD protocols to provide a key to the participant of symmetrical system transmission in a safe manner. Although there are several types of QKD protocols, they consist of nearly identical steps at a high level, but differ, among others, in the way the quantum particles or photons are prepared and transmitted over quantum channel [29,72]. Communication via the public channel is referred as *post-processing* and is used to extract the secret key from the raw key which is generated over the quantum channel. Although there are differences in implementations, almost every post-processing application needs to implement the following steps: the extraction of the raw key (sifting), error rate estimation, key reconciliation, privacy amplification and authentication. Given that the focus of the QKDNetSim is placed on network traffic and considering that there are different variations of the post-processing applications, QKDNetSim provides a simple application which seeks to imitate the traffic that is generated by the real-world post-processing applications such as AIT R10 QKD software [71]. The goal was to build an application that credibly imitates the traffic from the existing post-processing applications to reduce the simulation time and computational resources. Figure 6 shows the comparison of the traffic generated using QKDNetSim post-processing application and AIT R10 QKD Software. The Pearson correlation coefficients between the measurement and modelling results is 0.732.

It is important to emphasize that the impact of post-processing applications in a QKD network cannot be ignored. Considering that nodes in a QKD network constantly generate keys at their maximum rate until their key storages are filled [28], in some cases, the traffic generated by the post-processing application can have a significant impact on the communication over the public channel, especially when it comes to the public channels of weaker network performances.

3.3.7 QKD helper

QKDNetSim comes with a helper class (QKD helper) which provides an easier installation of QKD managers at network nodes, setting the parameters of QKD links and QKD buffers, and drawing the QKD graphs. Given that cryptographic operations are called from QKD managers, the QKD module can be easily used in overlay networks with two TCP/IP stacks or in simple networks with a single TCP/IP stack. When an overlay network is used, it is necessary that the applications use *virtual-TCP-L4* or *virtual-UDP-L4* protocols which pass the packet to the virtual-ipv4-l3 protocol. The packet is further passed to the corresponding NetDevice and QKD manager. After performing cryptographic operations, the packet is finally delivered to the underlying network. In the case of a simple network when a single TCP/IP stack is used, standard *ns3::tcp-l4*, *ns3::udp-l4* and *ns3::ipv4-l3* protocols are used, while the QKD helper sets sending and receiving ns3::callback from NetDevice to the QKD manager to perform cryptographic operations. The QKD helper is realized to facilitate the procedure for the establishment of such configurations.

3.4 Verification

Although the literature states that a variety of applications have been tested in previously deployed QKD networks [9,68,73–78], the research in the performance of public channels was generally underestimated and most attention was paid to the quantum channel performances. To the best of our knowledge, besides work reported in [62], no other reports on network traffic analysis over the public channel have been published. Additionally, although there are several software applications dealing with QKD [50–55], to the best of our knowledge, applications for simulating QKD networks with multiple nodes and links are not available.

Therefore, QKDNetSim is a unique tool that aims at enabling the publication of the new results and findings in a communication over a public channel of the QKD link. It is important to note that QKDNetSim is a simulation module for the QKD Network which is developed in a well-proven existing NS-3 simulator [13,59]. QKDNetSim uses the well-known Crypto++ open-source C++ class cryptographic library [70] and well-tested code libraries of NS-3 simulator with minimal modifications to allow overlay network communication and the packet encapsulation with the QKD Header. Given the popularity and utilization of the AIT R10 QKD software in previously implemented projects [9,14,71] QKDNetSim follows the network organization from the previously deployed QKD testbeds which is reflected in the design of the QKD

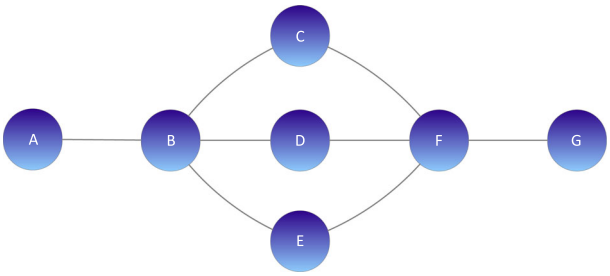

Fig. 7 Topology of a simulated single TCP/IP stack network in which the encrypted data UDP flow of rate 300 kbps between nodes *A* and *G* is established

Header and the imitation of the network traffic generated using AIT R10 QKD post-processing application[1] as described in Sect. 3.3.6.

3.5 Example of use

QKDNetSim can be utilized for a simple performance testing of routing protocols in the QKD Network. In our example, we set up a simulation with 7 fixed nodes forming the topology shown in Fig. 7 using a single TCP/IP network stack to test AODV, DSDV and OLSR routing protocols which are embedded in the NS-3 simulator by default. Table 2 presents the model parameters including the key generation rate, charging key rate, packet size, and data traffic parameters. The parameters not given here are the default parameters of the NS-3 simulator, version 3.26.

The parameters in Table 2 indicate that the links C–F, D–F, and E–F have the least amount of initial key material. As tested routing protocols have no information about the state of links, they need to choose between one of the three possible routes: A–B–C–F–G, A–B–D–F–G or A–B–E–F–G. Yet, after a while, the usage of any of these paths results in a disruption of communication since the available key material is quickly consumed and QKD links become unavailable. Then, the routing protocol in use needs to choose an alternative route while the depleted link is charged and recovered. When the used link is depleted again, the routing protocol should switch to an alternate path and so on. It is important to stress that links without available key material are unavailable only for the communication that requires the usage of key material but IPv4 interfaces remain active for any unencrypted communication. It means that the routing protocol which does not measure the state of QKD buffers is not able to instantly register link availability changes. On the other hand, given that the main objective of QKD is to provide ITS communication, routing packet needs to be encrypted and authenticated [14]. Therefore, in our simulation, each routing packet is encrypted and authenticated, and in the case, when there is no enough key material, routing packets are not transmitted. Consequently, the only way for routing protocol to detect the unavailability of the link is the detection of lack of routing

[1] The AIT QKD R10 is free for download w/o registration via git with "git clone http://sqt.ait.ac.at/git/qkd-public.git" [71].

Table 2 Parameter values of the simulation

Parameter	Value
Total number of nodes	6
Packet size	512 Bytes
Packet traffic type	UDP; CBR
Packet traffic rate	300 kbps
Encryption type	OTP
Authentication type	VMAC
Authentication tag length	32 bits
Maximal amount of key material (all links)	6400 kBytes
Minimal amount of key material (all links)	128 kBytes
Threshold value (all links)	2000 kBytes
Traffic queue capacity (per device)	10000 Packets
Initial amount of key material (link A–B)	6400 kBytes
Initial amount of key material (link B–C)	6400 kBytes
Initial amount of key material (link B–D)	6400 kBytes
Initial amount of key material (link B–E)	6400 kBytes
Initial amount of key material (link C–F)	1600 kBytes
Initial amount of key material (link D–E)	1032 kBytes
Initial amount of key material (link E–F)	750 kBytes
Initial amount of key material (link F–G)	6400 kBytes
Charging key rate for all links (per second)	12.8 kBytes
Total simulation time	300 s

packets and routing information. It is important to note that when QKD buffer is in the EMPTY state, that is when the amount of key material is below the minimal threshold, the residual key material should be only used for the establishment of new key material [29]. To realize such behaviour, we implemented priority waiting queues which perform classification of the network traffic using Differentiated Services Code Point (DSCP)/Type of Service (ToS) bits in the IP header. The traffic generated by QKDNetSim charging application has the highest priority while the traffic originated from routing protocol as well as user's traffic has the lower priority. Table 3 compares the obtained values based on the number of sent routing data and Packet Delivery Ratio (PDR) which is calculated as the ratio of received and sent application packets. These two values are used to assess the effectiveness of the routing protocol within the specified simulation environment.

Table 3 shows that AODV sends the largest number of routing packets. AODV is known as a reactive routing protocol where routing paths are searched only when needed by flooding the network [79,80]. The discovery procedure terminates when either a route has been found, or no route is available after all route permutations have been checked which results in a large key material consumption as shown in Fig. 8. In our simulation, AODV uses only the A–B–E–F–G route while other routes

Table 3 Comparison of the obtained values

Routing protocol	Routing data		Application data
	Packets	Bytes	PDR (%)
AODV (RFC 3561)	10,839	510,952	33.7772
AODV (modified)	12,887	611,480	68.9868
DSDV	4786	338,464	73.4228
OLSR	1620	125,328	49.9641

The number of routing packets and packet delivery ratio (PDR—which is calculated as the ratio of received and sent application packets) are used to assess the effectiveness of the routing protocol

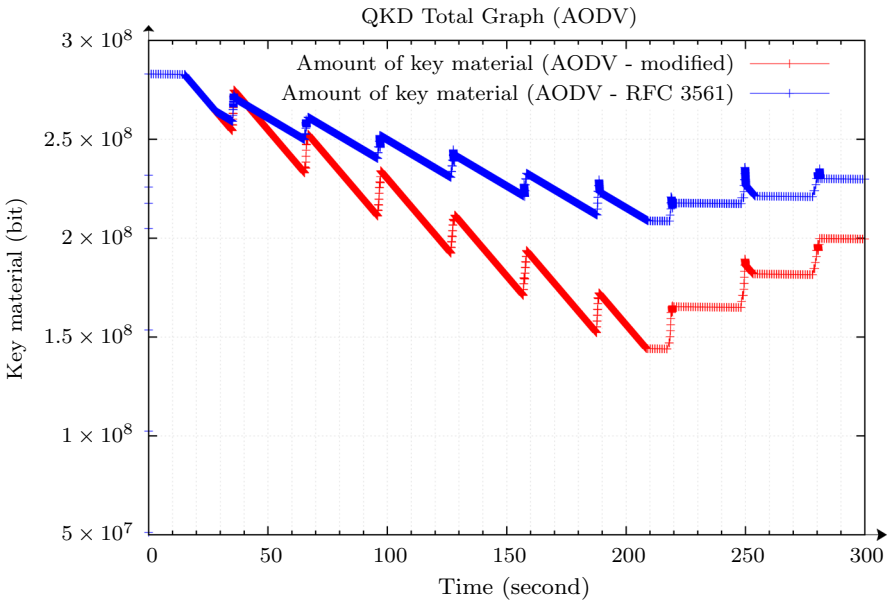

Fig. 8 The QKDNetSim total graph shows the total consumption and generation of key material in the network when the AODV routing protocol is used. In our simulation, the new material is generated each 30 s as shown in Fig. 6 resulting in a periodic increase in the available key material

are ignored. Given that AODV by default has no method for measuring the state of the links, it assumes A–B–E–F–G is the best route towards the destination. When the key material on the E–F link is depleted, the A–B–E–F–G path is unsustainable but no alternative route towards destination is found (shown in Fig. 8 in the period from 220–250 s).

According to RFC 3561, AODV needs to update the Active Route Life of the route which is used for packet forwarding to be no less than the current time plus ACTIVE_ROUTE_TIMEOUT which is set to 3 s by default. Therefore, the forwarding node is in charge to extend the validity of the route towards the destination regardless of whether that route is feasible at all. In our example, it means that node B assumes

that the route towards destination G is valid over node E even in the case when there is a lack of key material to forward the packet over the E–F link.

Each time the E–F link is recovered (QKD buffers are charged with the new key material), AODV on node E floods the route request for establishing of the route towards destination resulting in periodic peaks shown in Fig. 9 in a period of 0–220 s of the simulation time. To emphasize the impact of the policy in the AODV forwarding mode, we modified the forwarding policy to obey the updating of the route towards the destination in a forwarding mode. In this case, AODV sends routing packets to refresh the route each time when the route expires and communication with the destination is requested, but it does not update the route towards the destination in the forwarding mode at all. As shown in Fig. 9, our modification resulted in additional routing traffic but it improved the PDR for more than double. Also, our modification prevents futile consumption of the key material on links A–B and B–E in the case when no route towards the destination is feasible as shown in Fig. 8. In the period 220–300 s of the simulation time, the key material on links A–B, B–E and E–F is depleted and each time new key material is generated, AODV requests for the route towards destination G resulting in the large peaks as shown in Fig. 9. It is important to note that AODV maintains the obtained route as long as that route is used. Since QKD devices constantly generate keys at their maximum key rate until the key storages are filled [28], the routes to the neighbouring nodes are constantly maintained.

The DSDV proactive routing protocol uses the A–B–E–F–G route and switches to the A–B–D–F–G route when the key material on the E–F link is depleted. Thereafter, DSDV uses routes A–B–C–F–G and combines all three routes towards the destination. Given that DSDV detects a lack of routing information only after an exchange of routing tables which by default is set to 15 s (no triggered updates since IPv4 interfaces remain active) [81,82], the source node assumes that the route to the destination is available and sends the encrypted traffic which is silently discarded on intermediate nodes due to the lack of the available key material for further forwarding. In our simulation, the key material establishment process took about 30 s as shown in Fig. 6 while the periodic update interval of DSDV is set to 15 s by default. Hereof, in the process of key material establishment, DSDV noticed the lack of the periodic update from the node which is connected with a link with the depleted key material. DSDV deletes the entry in the routing table towards that node which interrupts the key material establishment process and leaves the link in the locked position. Due to the lack of a route, it is not possible to generate the new key material and due to lack of the key material, it is not possible to exchange routing packets to update routing tables. Such a scenario is registered for the C–F link which remains locked after second 160 and disables the A–B–C–F–G route.

OLSR is based on a proactive link-state approach which uses Hello and Topology Control (TC) routing messages to discover and disseminate link-state information throughout the network. OLSR reduces the control traffic overhead by using Multipoint Relays (MPR), which is the key idea behind OLSR. In our simulation, OLSR uses all three possible routes towards the destination. By default, OLSR exchanges Hello messages each 2 s and defines "holding time" as three times the Hello message period. Holding time indicates a waiting time prior deleting route due to lack of fresh Hello message [12]. Therefore, OLSR is able to detect a link failure after 6 s and if key

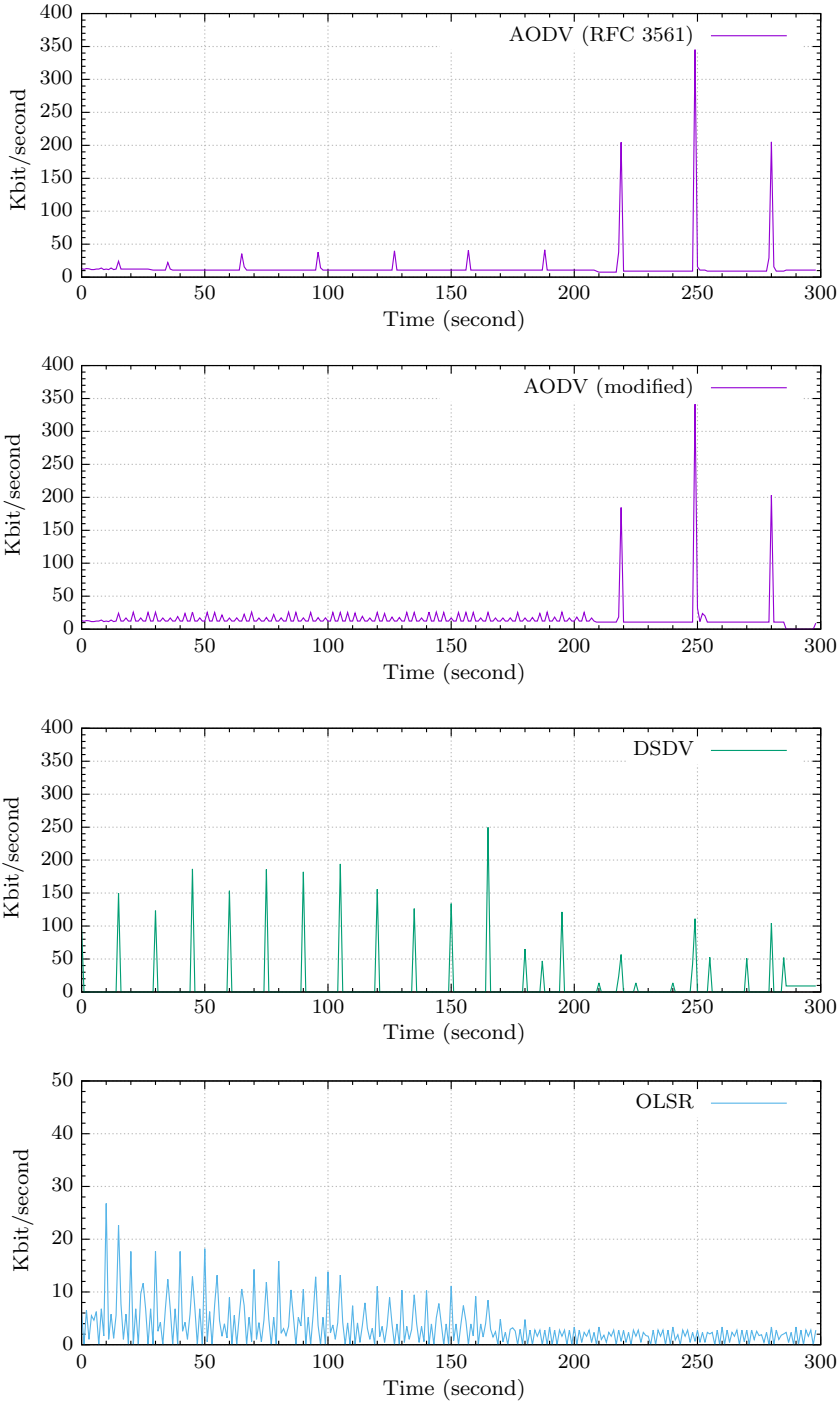

Fig. 9 Comparison of the traffic generated by the routing protocols

material establishment process takes longer, the probability that the link with the EMPTY state of QKD buffers will become locked increases.

4 Conclusion

This paper deals with the practical realization of QKD networks from a network point of view. The work summarizes the limitations and the basic characteristics of a QKD network and describes the ways of implementing QKD networks, which can be either in an overlay mode or as a network with a single TCP/IP stack. The main part of this paper deals with the QKDNetSim simulation environment which is primarily intended for testing of the existing solutions and the implementation of the new solutions in the field of QKD networks. An example of the use of QKDNetSim described in this document indicates that the traffic originated from the routing protocol needs to have a higher priority in the allocation of the network resources to avoid bringing QKD links in a locked position. Also, the obtained results confirm the results previously published in [12] stating that the DSDV routing protocol yields significantly better performance in terms of PDR and the number of the transmitted routing packets when compared to other tested routing protocols. Considering that currently there are no available simulators of QKD networks, QKDNetSim has significant benefits for the research community in the field of QKD technologies. We assume that QKDNetSim will facilitate understanding of the practical use of the QKD technology which, in turn, enables the use of a wide range of applications within the QKD network. Although primarily designed for the QKD network, QKDNetSim can be easily applied to simulate other types of networks. The implementation of a virtual-TCP/IP stack allows for a simple simulation of the overlay networks while QKD Buffers and QKD Cryptos simplify the simulations that involve the use of a symmetrical cryptographic key. The QKDNetSim source code is free for download from the *git* repository "https://bitbucket.org/liptel/qkdnetsim".

The main contribution of this paper is the presentation of a practical organization and simulation environment of the QKD network. Our future work will focus on the emulation of QKD networks.

Acknowledgements The authors are grateful to the anonymous reviewers for their comments and suggestions that helped improve the quality of this paper. The research received a financial support from the SGS Grant No. SP2017/174, VSB - Technical University of Ostrava, Czech Republic.

References

1. Bennett, C.H., Brassard, G.: Quantum cryptography: public key distribution and coin tossing. Theor. Comput. Sci. **560**, 7–11 (2011)
2. Alleaume, R., Branciard, C., Bouda, J., Debuisschert, T., Dianati, M., Gisin, N., Godfrey, M., Grangier, P., Langer, T., Lutkenhaus, N., Monyk, C., Painchault, P., Peev, M., Poppe, A., Pornin, T., Rarity, J., Renner, R., Ribordy, G., Riguidel, M., Salvail, L., Shields, A., Weinfurter, H., Zeilinger, A.: Using quantum key distribution for cryptographic purposes: a survey. Theor. Comput. Sci. **560**(P1), 62–81 (2014)

3. Maurer, U.: Information-theoretically secure secret-key agreement by NOT authenticated public discussion. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). **1233**, 209–225 (1997)
4. Bennett, C.H., Brassard, G.: Quantum cryptography: public key distribution and coin tossing. In: Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, vol. 175, p. 8. New York (1984)
5. Renner, R.: Security of quantum key distribution. Ph.D. thesis, Swiss Federal Institute of Technology Zurich (2005)
6. Alleaume, R., Bouda, J., Branciard, C., Debuisschert, T., Dianati, M., Gisin, N., Godfrey, M., Grangier, P., Langer, T., Leverrier, A., Lutkenhaus, N., Painchault, P., Peev, M., Poppe, A., Pornin, T., Rarity, J., Renner, R., Ribordy, G., Riguidel, M., Salvail, L., Shields, A., Weinfurter, H., Zeilinger, A.: SECOQC White Paper on Quantum Key Distribution and Cryptography, p. 28. arXiv preprint arXiv:quant-ph/0701168 (2007)
7. Elliott, C., Yeh, H.: DARPA quantum network testbed. Technical Report July, BBN Technologies Cambridge, New York, USA (2007)
8. Xu, F.X., Chen, W., Wang, S., Yin, Z.Q., Zhang, Y., Liu, Y., Zhou, Z., Zhao, Y.B., Li, H.W., Liu, D., Han, Z.F., Guo, G.C.: Field experiment on a robust hierarchical metropolitan quantum cryptography network. Chin. Sci. Bull. **54**(17), 2991–2997 (2009)
9. Sasaki, M., Fujiwara, M., Ishizuka, H., Klaus, W., Wakui, K., Takeoka, M., Miki, S., Yamashita, T., Wang, Z., Tanaka, A., Yoshino, K.: Field test of quantum key distribution in the Tokyo QKD network. Opt. Express **19**(11), 10387–10409 (2011)
10. Wang, S., Chen, W., Yin, Z.Q., Li, H.W., He, D.Y., Li, Y.H., Zhou, Z., Song, X.T., Li, F.Y., Wang, D., Chen, H., Han, Y.G., Huang, J.Z., Guo, J.F., Hao, P.L., Li, M., Zhang, C.M., Liu, D., Liang, W.Y., Miao, C.H., Wu, P., Guo, G.C., Han, Z.F.: Field and long-term demonstration of a wide area quantum key distribution network. Opt. Express **22**(18), 21739 (2014)
11. Yin, J., Cao, Y., Li, Y.H., Liao, S.K., Zhang, L., Ren, J.G., Al, W.Q.C., Liu, W.Y., Bo Li, H.D., Li, G.B., Lu, Q.M., Gong, Y.H., Xu, Y., Li, S.L., Li, F.Z., Yin, Y.Y., Jiang, Z.Q., Li, M., Jia, J.J., Ge Ren, D.H., Zhou, Y.L., Zhang, X.X., Wang, N., Chang, X., Zhu, Z.C., Liu, N.L., Chen, Y.A., Lu, C.Y., Shu, R., Peng, C.Z., Wang, J.Y., Pan, J.W.: Satellite-based entanglement distribution over 1200 kilometers. Science **356**(6343), 1140–1144 (2017)
12. Mehic, M., Fazio, P., Voznak, M., Chromy, E.: Toward designing a quantum key distrubution network. Adv. Electr Electron. Eng. **14**(4), 413–420 (2016)
13. Henderson, T.R., Riley, G.F.: Network simulations with the ns-3 simulator. In: Proceedings of Sigcomm, pp. 527 (2006)
14. Kollmitzer, C., Pivk, M. (eds.): Applied Quantum Cryptography. Lecture Notes in Physics, vol. 797. Springer, Berlin Heidelberg (2010). doi:10.1007/978-3-642-04831-9
15. Dianati, M., Alleaume, R.: Architecture of the Secoqc quantum key distribution network. In: 2007 First International Conference on Quantum, Nano, and Micro Technologies (ICQNM'07), IEEE, pp. 13–13, Jan 2007
16. Mehic, M., Maurhart, O., Rass, S., Komosny, D., Rezac, F., Voznak, M.: Analysis of the public channel of quantum key distribution link. IEEE J. Quantum Electron. (2017, in press)
17. Ciurana, A., Martinez-Mateo, J., Peev, M., Poppe, A., Walenta, N., Zbinden, H., Martin, V.: Quantum metropolitan optical network based on wavelength division multiplexing. Opt. Express **22**(2), 1576–93 (2014)
18. Aleksic, S., Winkler, D., Franzl, G., Poppe, A., Schrenk, B., Hipp, F.: Quantum key distribution over optical access networks. In: Proceedings of the 2013 18th European Conference on Network and Optical Communications and 2013 8th Conference on Optical Cabling and Infrastructure (NOC-OC&I), pp. 11–18. (2013)
19. Alleaume, R., Roueff, F., Diamanti, E., Lutkenhaus, N.: Topological optimization of quantum key distribution networks. N. J. Phys. **11**(7), 075002 (2009)
20. Gisin, N., Ribordy, G., Tittel, W., Zbinden, H.: Quantum cryptography. Rev. Mod. Phys. **74**(1), 145–195 (2002)
21. Salvail, L., Peev, M., Diamanti, E., Alléaume, R., Lütkenhaus, N., Länger, T.: Security of trusted repeater quantum key distribution networks. J. Comput. Secur. **18**(1), 61–87 (2010)
22. Scarani, V., Bechmann-Pasquinucci, H., Cerf, N.J., Dušek, M., Lütkenhaus, N., Peev, M.: The security of practical quantum key distribution. Rev. Mod. Phys. **81**(3), 1301–1350 (2009)

23. Dušek, M., Lütkenhaus, N., Hendrych, M.: Quantum cryptography. In: Progress in Optics, vol. 49, pp. 381–454 (2006). http://www.sciencedirect.com/science/article/pii/S0079663806490053
24. Dixon, A.R., Yuan, Z.L., Dynes, J.F., Sharpe, A.W., Shields, A.J.: Continuous operation of high bit rate quantum key distribution. Appl. Phys. Lett. **96**(2010), 2008–2011 (2010)
25. Korzh, B., Lim, C.C.W., Houlmann, R., Gisin, N., Li, M.J., Nolan, D., Sanguinetti, B., Thew, R., Zbinden, H.: Provably secure and practical quantum key distribution over 307 km of optical fibre. Nat. Photon. **9**(3), 163–168 (2015)
26. Sasaki, M.: Tokyo QKD network and the evolution to secure photonic network. In: CLEO:2011—Laser Applications to Photonic Applications, vol. 1., p. JTuC1. OSA, Washington, D.C. (2011)
27. Wang, S., Chen, W., Guo, J.F., Yin, Z.Q., Li, H.W., Zhou, Z., Guo, G.C., Han, Z.F.: 2 GHz clock quantum key distribution over 260 km of standard telecom fiber. Opt. Lett. **37**(6), 1008 (2012)
28. Dianati, M., All, R., Alléaume, R., Gagnaire, M., Shen, X.S.: Architecture and protocols of the future European quantum key distribution network. Secur. Commun. Netw. **1**(1), 57–74 (2008)
29. Mehic, M., Niemiec, M., Voznak, M.: Calculation of the key length for quantum key distribution. Elektron. ir Elektrotech. **21**(6), 81–85 (2015)
30. Elliott, C.: Building the quantum network. N. J. Phys. **4**, 346 (2002)
31. Abidin, A., Larsson, J.Å.:Security of Authentication with a Fixed Key in Quantum Key Distribution, p. 14 (2011)
32. Portmann, C.: Key recycling in authentication. IEEE Trans. Inf. Theory **60**(7), 4383–4396 (2014)
33. Hao, W., Zheng-Fu, H., Guang-Can, G., Pei-Lin, H.: The queueing model for quantum key distribution network. J. Phys. G **G36**(7), 25006 (2009)
34. Konig, S., Rass, S.: On the transmission capacity of quantum networks. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **2**(11), 9–16 (2011)
35. Rass, S., König, S.: Turning quantum cryptography against itself: how to avoid indirect eavesdropping in quantum networks by passive and active adversaries. Int. J. Adv. Syst. Meas. **5**(1), 22–33 (2012)
36. Collins, D., Gisin, N., de Riedmatten, H.: Quantum relays for long-distance quantum cryptography. J. Mod. Opt. **52**, 735–753 (2005)
37. Dur, W., Briegel, H.J., Cirac, J.I., Zoller, P.: Quantum repeaters based on entanglement purification. Phys. Rev. A **59**(1), 169–181 (1999)
38. Yuan, Z.S., Chen, Y.A., Zhao, B., Chen, S., Schmiedmayer, J., Pan, J.W.: Experimental demonstration of a BDCZ quantum repeater node. Nature **454**(7208), 1098–1101 (2008)
39. Poppe, A., Peev, M., Maurhart, O.: Outline of the SECOQC quantum-key-distribution network in Vienna. J. Quantum Inf. **6**(2), 10 (2008)
40. Sergienko, A.V. (ed.): Quantum Communications and Cryptography, Optical Science and Engineering, 1st edn. CRC Press, Boca Raton (2005)
41. Marhoefer, M.,Wimberger, I., Poppe, A.: Applicability of Quantum Cryptography for Securing Mobile Communication Networks (2006). https://pdfs.semanticscholar.org/53a8/4df168139553023c95f11967b0e2462bd6bd.pdf
42. Hayashi, M., Iwama, K., Nishimura, H., Raymond, R., Yamashita, S.: Quantum network coding. In: Thomas, W., Weil, P. (eds.) Lecture Notes in Computer Science. Volume 4393 of Lecture Notes in Computer Science, pp. 610–621. Springer, Berlin (2007)
43. Schartner, P., Rass, S., Schaffer, M.: Quantum key management. In: Applied Cryptography and Network Security. InTech (2012)
44. Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay networks. Sosp **32**(1), 66 (2001)
45. Rekhter, Y., Li, T., Hares, S.: A Border Gateway Protocol 4 (BGP-4), No. RFC 4271 (2005). http://www.rfc-editor.org/rfc/rfc4271.txt
46. Labovitz, C., Ahuja, A., Wattenhofer, R., Venkatachary, S.: The impact of internet policy and topology on delayed routing convergence. In: Proceedings IEEE INFOCOM 2001, Conference on Computer Communications, Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213), vol. 1, pp. 537–546. (2001)
47. Chun, B.G., Fonseca, R., Stoica, I., Kubiatowicz, J.: Characterizing selfishly constructed overlay routing networks. In: IEEE INFOCOM 2004, IEEE, vol. 2., pp. 1329–1339. (2004)
48. Lee, G.M., Choi, T.: Improving the interaction between overlay routing and traffic engineering. In: NETWORKING 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet, vol. 4982, pp. 530–541. LNCS, Springer, Berlin (2008)

49. Liu, Y., Zhang, H., Gong, W., Towsley, D.: On the interaction between overlay routing and underlay routing. In: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, vol. 4, pp. 2543–2553. (2005)
50. Niemiec, M., Romanski, L., Swiety, M.: Quantum cryptography protocol simulator. In: Communications in Computer and Information Science, CCIS, vol. 149, pp. 286–292. (2011)
51. Pereszlenyi, A.: Simulation of quantum key distribution with noisy channels. In: Proceedings of the 8th International Conference on Telecommunications, ConTEL, IEEE , vol. 1, pp. 203–210. (2005)
52. Zhang, X., Wen, Q.: Object-oriented quantum cryptography simulation model. In: Third International Conference on Natural Computation, Number ICNC, IEEE, pp. 7–10. (2007)
53. Zhao, S., De Raedt, H., Liu, B., Huang, Y.: Event-by-event simulation of quantum cryptography protocols. J. Comput. Theor. Nanosci. **5**(7), 1251–1254 (2008)
54. Buhari, A.: An efficient modeling and simulation of quantum key distribution protocols using OptiSystem. In: IEEE Symposium on Industrial Electronics and Applications (ISIEA), pp. 84–89. Bandung (2012)
55. Mailloux, L.O., Morris, J.D., Grimaila, M.R., Hodson, D.D., Jacques, D.R., Colombi, J.M., Mclaughlin, C.V., Holes, J.A.: A modeling framework for studying quantum key distribution system implementation nonidealities. IEEE Access **3**, 110–130 (2015)
56. Breslau, L., Estrin, D., Fall, K., Floyd, S., Heidemann, J., Helmy, A., Huang, P., McCanne, S., Varadhan, K., Xu, Y., Yu, H.: Advances in network simulation. Computer **33**(5), 59–67 (2000)
57. Cavin, D., Sasson, Y., Schiper, A.: On the accuracy of MANET simulators. In: Proceedings of the second ACM International Workshop on Principles of Mobile Computing—POMC '02, vol. 1, p. 38. ACM Press, New York (2002)
58. Andel, T., Yasinac, A.: On the credibility of manet simulations. Computer **39**(7), 48–54 (2006)
59. Altman, E., Jimenez, T.: NS Simulator for Beginners. In: Walrand, J. (ed.) Synthesis Lectures on Communication Networks, vol. 5. Morgan and Claypool Publishers (2012). doi:10.2200/S00397ED1V01Y201112CNT010
60. Aguado, A., Martin, V., Lopez, D., Peev, M., Martinez-Mateo, J., Rosales, J., de la Iglesia, F., Gomez, M., Hugues-Salas, E., Lord, A., Nejabati, R.: Quantum-aware software defined networks. In: 6th International Conference on Quantum Cryptography (QCRYPT 2016), p. 3. QCrypt, Washington, DC (2016)
61. Rass, S., Sandra, K.: Indirect Eavesdropping in Quantum Networks. In: The Fifth International Conference on Quantum, Nano and Micro Technologies, ICQNM, pp. 83–88. (2011)
62. Mehic, M., Komosny, D., Mauhart, O., Voznak, M., Rozhon, J.: Impact of packet size variation in overlay quantum key distribution network. In: 2016 XI International Symposium on Telecommunications (BIHTEL), Sarajevo, Bosnia and Herzegovina, IEEE, pp. 1–6. (2016)
63. Cederlöf, J.: Authentication in quantum key growing. Ph.D. thesis, Master Thesis, Linköping University (2005)
64. Wehrle, K., Günes, M., Gross, J. (eds.): Modeling and Tools for Network Simulation, 1st edn. Springer-Verlag Berlin Heidelberg (2010). doi:10.1007/978-3-642-12331-3
65. Riley, G.F., Henderson, T.R.: The ns-3 network simulator. In: Modeling and Tools for Network Simulation, pp. 15–34. Springer, Berlin (2010)
66. Fall, K., Varadhan, K.: The Network Simulator (ns-2). http://www.isi.edu/nsnam/ns (2007)
67. Peev, M., Pacher, C., Alléaume, R., Barreiro, C., Bouda, J., Boxleitner, W., Debuisschert, T., Diamanti, E., Dianati, M., Dynes, J.F., Fasel, S., Fossier, S., Fürst, M., Gautier, J.D., Gay, O., Gisin, N., Grangier, P., Happe, A., Hasani, Y., Hentschel, M., Hübel, H., Humer, G., Länger, T., Legré, M., Lieger, R., Lodewyck, J., Lorünser, T., Lütkenhaus, N., Marhold, A., Matyus, T., Maurhart, O., Monat, L., Nauerth, S., Page, J.B., Poppe, A., Querasser, E., Ribordy, G., Robyr, S., Salvail, L., Sharpe, A.W., Shields, A.J., Stucki, D., Suda, M., Tamas, C., Themel, T., Thew, R.T., Thoma, Y., Treiber, A., Trinkler, P., Tualle-Brouri, R., Vannel, F., Walenta, N., Weier, H., Weinfurter, H., Wimberger, I., Yuan, Z.L., Zbinden, H., Zeilinger, A.: The SECOQC quantum key distribution network in Vienna. N. J. Phys. **11**(7), 075001 (2009)
68. Dodson, D., Fujiwara, M., Grangier, P., Hayashi, M., Imafuku, K., Kitayama, K.I., Kumar, P., Kurtsiefer, C., Lenhart, G., Luetkenhaus, N., et al.: Updating quantum cryptography report ver. 1. arXiv preprint arXiv:0905.4325 (2009)
69. Cederlöf, J.: Larsson, Ja: Security aspects of the authentication used in quantum cryptography. IEEE Trans. Inf. Theory **54**(4), 1735–1741 (2008). doi:10.1109/TIT.2008.917697
70. Dai, W.: Crypto++ library. Retrieved from http://www.cryptopp.com (2017)

71. Maurhart, O., Pacher, C., Happe, A., Lor, T., Tamas, C., Poppe, A., Peev, M.: New release of an open source QKD software: design and implementation of new algorithms, modularization and integration with IPSec. In: Qcrypt 2013. (2013)
72. Mehic, M., Partila, P., Tovarek, J., Voznak, M.: Calculation of key reduction for B92 QKD protocol. In Donkor, E., Pirich, A.R., Hayduk, M., (eds.) SPIE Sensing Technology + Applications, International Society for Optics and Photonics, p. 95001J. (2015)
73. Fedrizzi, A., Poppe, A., Ursin, R., Lorünser, T., Peev, M., Länger, T., Zeilinger, A.: Practical quantum key distribution with polarization entangled photons. In: 2005 European Quantum Electronics Conference, EQEC '05, vol. 16, p. 303. (2005)
74. Stavroulakis, P., Stamp, M.: Handbook of Information and Communication Security. Springer, Heidelberg (2010)
75. Mink, A., Tang, X., Ma, L., Nakassis, T., Hershman, B., Bienfang, J.C., Su, D., Boisvert, R., Clark, C.W., Williams, C.J.: High speed quantum key distribution system supports one-time pad encryption of real-time video. Proc. SPIE **6244**, 62440M-1–7 (2006)
76. Tajima, A., Tanaka, A., Maeda, W., Takahashi, S., Tomita, A.: Practical quantum cryptosystem for metro area applications. IEEE J. Sel. Top. Quantum Electron. **13**(4), 1031–1037 (2007)
77. Mirza, A., Petruccione, F.: Realizing long-term quantum cryptography. J. Opt. Soc. Am. B **27**(6), A185 (2010)
78. Langer, T.: The practical application of quantum key distribution. Ph.D. thesis, University of Lausanne (2013)
79. Chakeres, I.D., Belding-Royer, E.M.: AODV Routing Protocol Implementation Design. In: Proceedings of 24th International Conference on Distributed Computing Systems Workshops, Hachioji, Tokyo, Japan, 23–24 March 2004, pp. 698–703. IEEE (2004)
80. Fazio, P., De Rango, F., Sottile, C.: An on demand interference aware routing protocol for VANETS. J. Netw. **7**(11), 1728–1738 (2012)
81. He, G.: Destination-Sequenced Distance Vector (DSDV) Protocol, pp. 1–9. Networking Laboratory, Helsinki University of Technology (2002). http://www.netlab.tkk.fi/opetus/s38030/k02/Papers/03-Guoyou.pdf
82. Mehic, M., Fazio, P., Voznak, M., Partila, P., Komosny, D., Tovarek, J., Chmelikova, Z.: On using multiple routing metrics with destination sequenced distance vector protocol for MultiHop wireless ad hoc networks. Int. Soc. Opt. Photon. 98480F (2016)