IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 September 2022, accepted 5 October 2022, date of publication 13 October 2022, date of current version 21 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3214511

RESEARCH ARTICLE

Tackling Denial of Service Attacks on Key Management in Software-Defined Quantum Key Distribution Networks

MIRALEM MEHIC[1,2], (Member, IEEE), STEFAN RASS[2,3,4], EMIR DERVISEVIC[1], AND MIROSLAV VOZNAK[2], (Senior Member, IEEE)

[1]Department of Telecommunications, Faculty of Electrical Engineering, University of Sarajevo, 71000 Sarajevo, Bosnia and Herzegovina
[2]VSB–Technical University of Ostrava, 708 00 Ostrava, Czechia
[3]LIT Secure and Correct Systems Laboratory, Johannes Kepler University Linz, 4040 Linz, Austria
[4]Institute of Artificial Intelligence and Cybersecurity, Universitaet Klagenfurt, 9020 Klagenfurt, Austria

Corresponding author: Miralem Mehic (miralem.mehic@ieee.org)

This work was supported in part by the Ministry of the Interior of the Czech Republic within the project Network Cybersecurity in Post-Quantum Era under Grant VJ01010008; in part by the EU Horizon 2020 Framework Programme (H2020) Project Open European Quantum Key Distribution Testbed (OPENQKD) under Agreement 857156; and in part by the Ministry of Science, Higher Education and Youth of Canton Sarajevo, Bosnia and Herzegovina, under Grant 27-02-11-41251-13/21, Grant 27-02-35-35137-29/22, and Grant 27-02-35-35143-6/22.

ABSTRACT A QKD network provides an additional security layer for IT-secure cryptographic key distribution that is added to existing conventional networks. Thus, QKD network components must be resilient to security challenges from conventional network environments. This paper provided a novel solution for designing a Key Management System resistant to DoS attacks. Our solution allows applications to function securely in environments with fewer keys. In addition, we have provided approaches for allocating and managing QKD resources to avoid malicious key reservations. Simulation experiments verified the proposed solutions.

INDEX TERMS Quantum key distribution networks, quality of service, simulations, networking.

I. INTRODUCTION

Quantum Key Distribution (QKD) employs physical laws to establish cryptographic keys between distant parties in an information-theoretically secure (ITS) manner [1]. As interest in QKD technology grows through practical testbeds, so does interest in security aspect of the commercially available equipment [2], [3].

QKD network stands out compared to conventional telecommunication networks in several aspects. One of which being the method of implementing QKD connections. The QKD link has two channels: a quantum channel for transmitting cryptographic values encoded in specific photon characteristics, and a public channel for verifying and processing the exchanged data. Each quantum channel is always a point-to-point link between exactly two nodes. Public channels, on the other hand, can be implemented as any ordinary connection with an arbitrary number of intermediary devices [4].

Given that current generation quantum systems produce keys at rates of up to several hundred kbps [5], this is mostly insufficient to support high-demand communication flows, emphasizing the necessity for effective key management. As a result, both endpoints of the corresponding link implement key buffers (storages) of limited capacity, which are gradually filled at their maximum key rate with the processed cryptographic key, referred to below as key material, and then used for data flow encryption/decryption [6]. The term "key material" refers to confidential shared secret bits created during the QKD process.

Such approach is adequate if direct point-to-point links are considered. However, as QKD systems develop commercially, there is a growing interest in implementing QKD technology in networks with a significantly larger number of nodes. For these purposes, it is necessary to implement dedicated key management systems (KMS) that will analyze

The associate editor coordinating the review of this manuscript and approving it for publication was Jesus Felez.

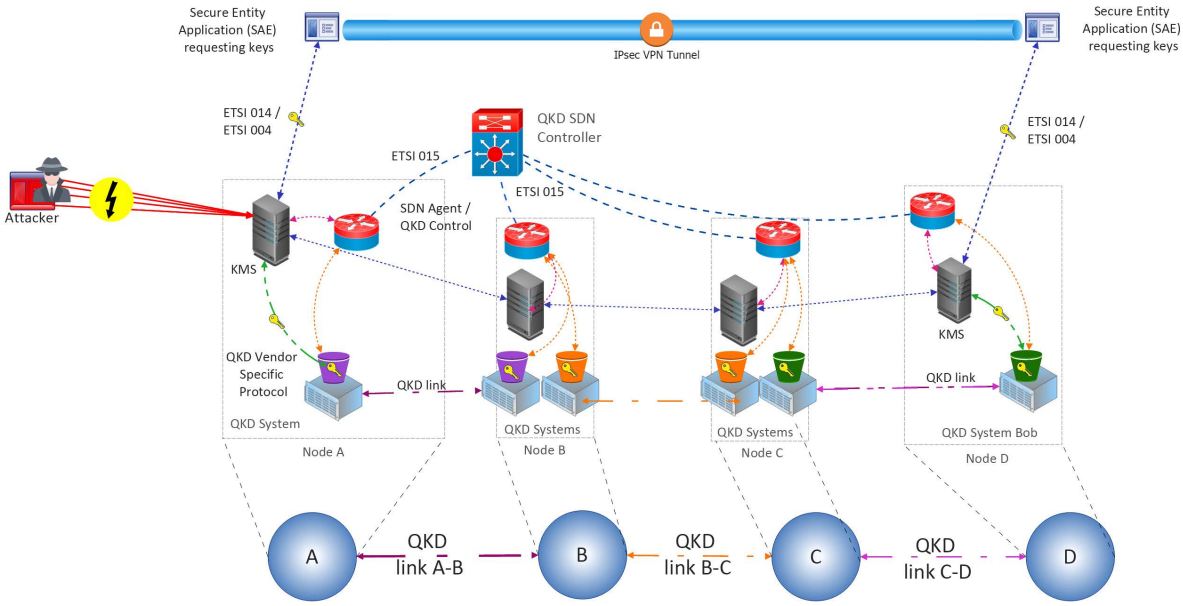

FIGURE 1. Logical topology of QKD network. The nodes forming the network include QKD systems, local KMS entites and QKD control. SDN QKD controller can be installed on a dedicated network node; it's logical position is neglected here. In this paper, we consider the influence of attackers on KMS with the aim of disabling functional work.

the requirements of user's applications, and based on the assessment and according to capabilities, provide these applications with the requested key material.

Each node in a QKD network based on trust-relay paradigm performs routing and forwarding operations [5], [7]. As a result, the organization of these networks was seen as distributed, with no hierarchical parent node. Thus, each node defines the best key delivery route based on information gathered from surrounding nodes. Nonetheless, QKD technology's proclivity to merge with existing telecom internet-service providers (ISP) networks is increasingly supporting a hierarchical strategy. Within the single domain, a local KMS (LKMS) can be implemented to serve local application requirements.

Since KMS stands out as the first point of contact for communication with user applications, it is reasonable to examine the security capabilities of this network component to disable the network. In this paper, we deal with the analysis of denial of service attacks on KMS entities. We use the QKDNetSim NS-3 network simulator to simulate network attacks, collect measurement results and evaluate the defense techniques [8].

The main contribution of this paper is proposing new approaches to allocating and managing resources in the QKD network to avoid DDoS attacks. To this end, the paper is organized as follows: Section II provides the overview of QKD network architecture, while section III provides information on the ETSI 004 key acquisition specification considered in this paper. In section IV, we provide theoretical and analytical guidelines on defending against malicious attacks. Section V describes the simulation experiment setup. The obtained results are discussed in section VI while section VII concludes this study and outlines the future work.

II. QKD ARCHITECTURE

Although there are several different approaches to defining the structure of the QKD node and thus defining the organization of the KMS entity, it is important to highlight the approach being developed within the EU H2020 OPENQKD[1] project [9].

Consider end-user applications that want to establish secure communication such as an IPsec VPN tunnel as shown in Fig. 1. To realize this communication, applications need to obtain previously exchanged secret keys [6]. According to the proposed structure the KMS is the initial point of contact for processing requests from key-seeking apps. It is coupled to a QKD control entity, which may operate and monitor the QKD system by executing typical activities like power-on/off, reboot, restart, QBER/temperature monitoring, and more. The KMS is aware of the status of QKD systems that create QKD link with surrounding nodes and can interact with other KMS entities to share management information.

Furthermore, the node's LKMS can connect with a specific routing entity, which attempts to compute the optimal route for distribution of keys based on information gathered about QKD link statuses and application needs. It is critical to emphasize that a routing/forwarding entity should be physically placed separately of the KMS entity, as this would give a great degree of flexibility. Thus, routing decisions can be made by external entities such as software-defined network (SDN) applications communicating with controllers. The separation of the routing component enables the network administrator to change or upgrade the routing logic as needed without interfering with communications between other entities in the network node.

[1] www.openqkd.eu

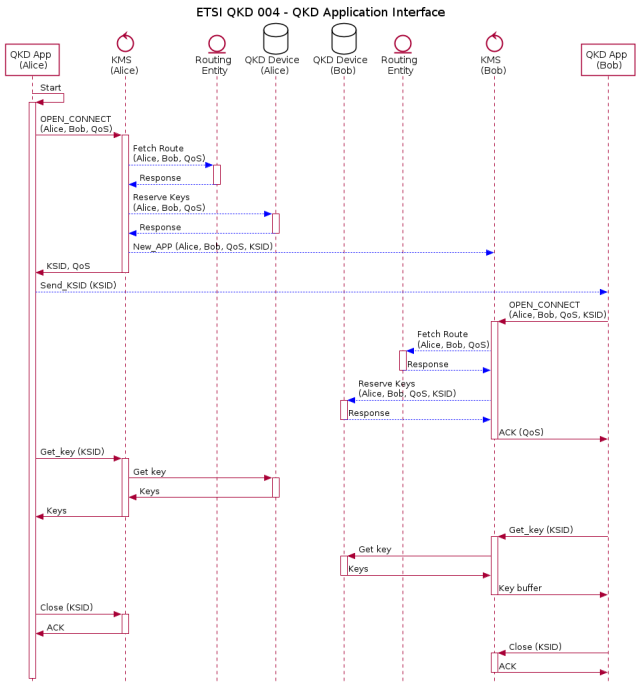

FIGURE 2. Sequence diagram of an ETSI 004 Application Interface exchanging QoS specifications. The blue dotted lines are outside the scope of the ETSI 004 standard [10].

III. QKD END-USER STANDARDS

There are several available standards that discuss QKD integration with IP networks. The ETSI 004 [10], which describes an API between end-user applications and QKD network components is highlighted in this paper. In a session-oriented paradigm, sessions are uniquely identified by a Key Session Identifier (KSID) value. If sessions records are not removed from the KMS memory, a session that has expired due to inactivity can be revived by using the same KSID value in the query.

The ETSI 004 standard defines three API functions:

- `OPEN_CONNECT`: a function for initiating a key stream session and reserving keys based on QoS factors. The input values for this message are the source and destination SAE IDs, as well as the QoS requirements. The answer contains the KSID as well as the status values.
- `GET_KEY`: a method for obtaining previously reserved keys. This message conveys the KSID identification, optional information, and key positioning index as input values (discussed further below). The response is a message that includes the key stream, optional metadata, output status, and information about the request's success.
- `CLOSE`: a function that terminates the key stream session and releases previously reserved resources. The KSID identification is used as the message's input value, which results in a status message that reports on the request's success.

As illustrated in Figure 2, Alice's SAE application attempts to interact with Bob's SAE application and sends the `OPEN_CONNECT` message to the nearest KMS system which is located within the same secure perimeter (area). This message seeks to reserve QKD keys on both ends of the future communication connection, with the option of specifying the minimal QoS condition. This call is stopped until either a connection is established or the timeout specified in the QoS-defined requirements for connection establishment (timeout) expires.

After receiving the `OPEN_CONNECT` request, the KMS will consider accepting the request and will contact the remote KMS to make the reservation. It should be noted that ETSI 004 does not indicate how to interact between KMS entities (blue dotted lines in Fig.2).

As a result, a KMS can make a reservation while waiting for a response from a distant KMS, or it can reply to SAE without waiting for a response from a remote KMS. It is important to note that in the first case, waiting for a response may cause additional delay, whereas in the second case, a collision may occur due to insufficient synchronization between the KMS entities (the KMS on Alice's side claims to have enough resources to accept the reservation request, but the KMS on Bob's side reports no available resources). This topic becomes especially important when contemplating a chain with numerous KMSs.[2]

Following our previous work [2], we propose the categorization of key requests. To avoid blocking work due to the lack of the key material used to generate new key material, the traffic generated by the post-processing application has the highest priority.[3] It is sorted in the top-priority (premium) queue. Signaling or routing packets within the network that are necessary for the smooth operation of network devices belong to a lower priority class. Additionally, several queues can be implemented for user data traffic that can be categorized into other priorities.

In ESTI 004, the notion is that the answer of `GET_KEY` is the synchronized key. Either the synchronized keys or an error message will be sent to the program. Although applications that communicate with the KMS should be authorized using certificates or dedicated security system mechanisms, the KMS still needs to implement a defense mechanism against malicious applications. Consider a situation where an attacker physically comes into possession of user terminal equipment with which he can communicate freely with the KMS. Certificates on the stolen user equipment are valid, the equipment is within a secure perimeter of communication, and there are no physical barriers that could prevent communication with KMS.

An attacker may intentionally obtain or reserve a large number of keys through the KMS to deplete available key

[2] In a trusted-repeater QKD network, a serving KMS is unaware of the available resources of KMSs on a path to a destination. As a result, it is difficult to put any guarantees that the requested level of service (QoS) can be fullfield until the capabilities of the path as a whole are confirmed.

[3] The QKD is a "key-expansion" scheme or "key-growing" technique because it requires a pre-shared key for authentication during the QKD-post-processing stage. As a result, a small portion of the newly generated (grown) key is always required to authenticate the following QKD processes.

resources and potentially influence routing mechanisms to redirect traffic to those links or devices under his control [11], [12]. In this context, an attacker will generate a large number of GET_KEY queries to obtain as many keys as possible in a short time. It is important to note that in the context of the ETSI 004 specification, it is common to expect an increased number of KSID sessions from the same IP address. Especially in cases when encryptors try to maintain a large number of IPsec sessions for which they initiate separate ETSI 004 sessions [6]. We analyzed such an approach in our previous work [13].

However, the more damaging attack is one where the attacker generates a large number of OPEN_CONNECT queries intending to reserve keys and thus deny the availability of keys for legitimate applications.

The KMS is left with several options for response:

1) ignores malicious queries. However, this approach requires a reliable algorithm for separating malicious key reservation requests from legitimate user queries. If the reliability of such algorithm cannot be guaranteed, it means that even legitimate queries for the key will be mislabeled and ignored.
2) responds to all queries (or only the first query) by sending feedback when too many OPEN_CONNECT requests are received from the same IP address. The KMS should implement firewall protection to stop the attacker's intentions, especially in the case of a DDoS attack from several different IP addresses. Also, the reliability of the firewall algorithm must be high to distinguish malicious from legitimate requests.
3) responds to queries according to priority so as not to fully reserve the requested amount of keys instantly (even if there are enough resources available). Higher priority requests will be served first, while KMS and SDN can categorize suspected malicious requests into lower priority queues. By providing a small portion of the key compared to the requested one, KMS aims to verify the credibility of the application's behavior. Applications that do not request new sessions before the expiration of previously approved ones can be considered credible and can be assigned more keys in subsequent iterations. Yet, to maintain a balance between key consumption and attainable security, KMS provides feedback to inform the requesting application that the requested amount of key material cannot be fully met. But, the transmission can be handled by resorting to less than perfect secrecy, i.e., a full one-time pad (OTP), to provide the requested bandwidth instead. Using SDN, it is possible to accomplish an adjustable balance between bandwidth and secrecy level to mitigate DoS upon too much key material requested. We describe this mechanism in section IV.

IV. BALANCING THE QoS AGAINST DoS

Information-theoretic confidentiality is the main selling point of having QKD networks. A less celebrated yet perhaps even more important aspect that QKD networks offer is their automated and transparent key management. While perfect secrecy via one-time pads may be a theoretical optimum, practical applications (excluding military contexts here), may allow a lower level of security, as long as bandwidth demands remain satisfiable. This is where DoS resilience becomes an issue, and to avoid the strategy of simply ignoring requests (which already is just another form of a service denial), it is possible to have the KMS cooperate with the tranmission system of the QKD network towards balancing the amount of requests against the amount of key material "growing back" per time unit.

It is useful to recall that OTP is perfectly secure, but has a key demand that is proportional (in fact equal) to the size of the transmitted payload. On the contrary, conventional AES (symmetric encryption) works with a fixed key size, but is only computationally secure. If the number of incoming requests is so large that a consumption proportional to the payload is insufficient or would let the key buffers run dry, the KMS and payload delivery control may switch to conventional means of AES, whose key demand is constant, and work with "only" computational secrecy, until the key-buffers have refilled. That is, if the DDoS or other attacks on KMS are about to exhaust the key buffers, it needs to use an alternative approach to provide service until KMS is back operational again, until the DDoS attack can be mitigated by other means.

This fallback option has been discussed in past literature [14], but never put to a practical experimental evaluation. The idea of randomly switching between OTP and AES has been studied in [15], and proven to retain a flexible level of secrecy-bandwidth-tradeoff by leveraging linear transformations. We repeat the idea here conceptually, to connect the relevant variables with the parameters that ETSI 015 specifies, for a practical implementation and evaluation of this concept.

Let the requested key material refer to a payload m of length $|m|$. Fix a value $n = 2^r$ as some integer power of 2, and split the message into blocks of size $\ell = \lceil \frac{|m|}{n} \rceil$ that fits into the *chunk size* of the network. This is the size of an encrypted packet. Practically, we would conversely look at the chunk size, and from that, choose $n = 2^r$ large enough that ℓ bits of payload fit into the chunk size.

Among the n payload blocks, let the end-user application make a random choice of k blocks that will be transmitted by using OTP, and the remaining $n-k$ blocks to be transmitted by conventional AES (e.g., in Galois/Counter mode, or similar). The point is that only a fraction of the payload will undergo the perfect OTP-protection and this reduces the demand for key material.

To "extend" the strength of the OTP over the k blocks towards all $n > k$ blocks, we apply a linear transformation, similar to an *all-or-nothing transformation* (to which it differs by the fact that our transformation matrix is fixed and publicly known). Postponing the concrete construction until Section IV-A, the idea is as follows: We choose an invertible matrix $\mathbf{A}$ with all nonzero entries over some Galois-field,

and (reversibly) map the message blocks to field elements $(m_1, \ldots, m_n) \in GF^n$. This vector is multiplied by $\mathbf{A}$ to give the vector $(c_1, \ldots, c_n)^\top = \mathbf{A} \cdot (m_1, \ldots, m_n)^\top$, in which each c_i then explicitly depends on all $m_1, \ldots, m_n$, so that none of the plaintext blocks is recoverable unless all of the c_i are in the adversary's possession; we call this process *mixing*.[4] It follows that the AES- and OTP-protected blocks "mutually protect" one another, and even an attacker with infinite computational power would be left with a residual uncertainty of $k \cdot \ell$ bits about the payload if all AES-concealed blocks were broken.[5] Since $\mathbf{A}$ is invertible, the recovery of the message blocks is by a multiplication with the inverse $\mathbf{A}^{-1}$. An efficient construction of a suitable matrix $\mathbf{A}$ is given in section IV-A.

If we measure the transmission's secrecy level by the fraction of Shannon-entropy before and after intercepting the message, we have $H(m_1, \ldots, m_n) = H(m_1) + H(m_2) + \ldots + H(m_n) = n \cdot \ell$ as the information transmitted (here assuming stochastic independence of message blocks), and entropy $H(m_1, \ldots, m_n | c_1, \ldots, c_n) = n \cdot \ell - (n-k) \cdot \ell = k \cdot \ell$ remaining unknown bits, assuming the worst case of a broken AES. The secrecy level can then be measured by the quantity

$$\text{secrecy-level} = \frac{H(m_1, \ldots, m_n | c_1, \ldots, c_n)}{H(m_1, \ldots, m_n)} = \frac{k}{n}, \quad (1)$$

which gives the

$$\text{required number of chunk key bits} = \min\{1 + k, n\} \cdot \ell \quad (2)$$

in which n is the size of the payload, and k is the number of blocks designated for OTP protection + 1 AES key. By varying k, we can "interpolate" between the two extremes of *perfect secrecy with the maximum chunk key bits* $n \cdot \ell = |m|$ bits, and *computational secrecy with the minimum number of chunk keys*, requiring $O(1)$ bits of key for the conventional AES.

The transmission is then by the following procedure: on input of n message blocks of size ℓ, transmit them in the following way, providing the above secrecy-level (eq. (1)) and key-demand (eq. (2)):

1) Apply a linear transformation to the overall payload to introduce mutual dependencies between all blocks (see Section IV-A for details)
2) Choose a random sample of k blocks to transmit via OTP, and encrypt the remaining blocks using AES
3) Upon reception, decipher the incoming blocks and invert the previous mixing transform (see Section IV-A)

[4] To distinguish it from cryptographic all-or-nothing transformations as being mostly understood as randomized invertible mappings, used as block-cipher modes [16].

[5] Practically, however, AES is believed to be resilient against attacks with quantum computers, for which Grover's algorithm would "merely" allow to half the search space, thus degrading AES-256 down to AES-128, but no substantial gain beyond this possibility is known as of the time of writing this paper.

A. THE MIXING TRANSFORM

Take any Galois field $GF(p^\ell)$ of odd characteristic p, and recall that such a field has elements being polynomials of degree $\ell - 1$. This enables a canonic embedding of a message block $\tilde{m}_i = b_0 b_1 \ldots b_{\ell-1} \in \{0,1\}^\ell$ into a field element as $\tilde{m}_i(X) = b_0 + b_1 X + b_2 X^2 + \ldots b_{\ell-1} X^{\ell-1} \in GF(p^\ell)$.

We construct the matrix $\mathbf{A}$ as a Hadamard-matrix, using Sylvester's recursive method: start from $\mathbf{A}_1 := (1)$ and define $\mathbf{A}_{2i} := \begin{pmatrix} \mathbf{A}_i & \mathbf{A}_i \\ \mathbf{A}_i & -\mathbf{A}_i \end{pmatrix}$ for $i = 1, 2, \ldots, r$. After r steps, this gives a matrix of dimension $2^r \times 2^r = n \times n$, which corresponds exactly to the number of blocks that we split the payload into (see above). The resulting matrix $\mathbf{A} = \mathbf{A}_n$ has (i) all entries either $+1$ or -1, and (ii) allows for an efficient inversion due to the identity $\mathbf{A}_n \cdot \mathbf{A}_n^\top = n \cdot \mathbf{I}$, so that $(m_1, \ldots, m_n)^\top = \frac{1}{n} \cdot \mathbf{A}^\top \cdot (c_1, \ldots, c_n)^\top$. The use of an odd prime as the characteristic is explained by this construction needing distinct elements $+1$ and -1, which would be identical in fields of characteristic 2.

B. HANDLING A DoS-DANGER BY KEY STORE EXHAUSTION ATTEMPTS

In situations where there is insufficient key material available, the end-user application may take different approaches to secure traffic using available cryptographic keys. But, in such critical situations, the network also has to consider ways of handling requests for keys.

Suppose that the one or more applications demand a total bandwidth of N bits per second, whereas the underlying QKD protocols can generate a lot of λ bits per second. The above transmission procedure starts by splitting the overall payload of N bits into blocks of size ℓ that is less than the chunk size, so that $n = N/\ell$ is an integer power of 2. Then, to avoid a DoS by overloading the network's key-regenerative capabilities, the KMS can choose k such that the key-demand is less than what the network can generate per second, and per application. That is, for a single application seeking to transmit N bits of payload, the KMS can pick the largest $k \leq n - 2$ such that

$$\text{requested key size } K = \min\left\{1 + k, \frac{N}{\ell}\right\} \cdot \ell \cdot \lceil \log_2(p) \rceil < \lambda, \quad (3)$$

whose variables are in direct correspondence to the parameters that ETSI 015 specifies. The factor $\log_2(p)$ comes in to capture the overhead of encoding field elements of $GF(p^\ell)$, where p is an odd prime (and hence ≥ 3, thus requiring at least 2 bits, making the key demand for a transmission of ℓ bits equal to $2 \cdot \ell$).

Conceptually, the choice of k is such that the key consumption becomes less than the key regeneration rate, so as to avoid the buffers to run empty and hence avoid DoS attack. From the security perspective, the tradeoff is secrecy vs. bandwidth, but the overall end-to-end security remains intact, by prior research results [15].

Following the previously discussed categorization of requests into priority queues [13], we calculated value of k as follows, and using the smallest possible characteristic $p = 3$, using 2 bits per element of $\mathbb{Z}_3$, which we use in the linear mixing (via Hadamard matrices).

To avoid exhaustion of the key storage with reservation requests by low-priority applications, SDN can set the parameter f that defines the threshold value (in percentage) as the ratio between the average consumption and effective secret key rate (eskr). KMS calculates value k as follows:

1) **high-priority requests:** KMS provides answers to high-priority requests by randomly selecting value k from range $[\lambda/(2K), \lambda/K]$.
2) **low-priority requests:** When the eskr b is $> f$, and request of low priority is received, KMS will define value k randomly from range $[1, \lambda/(2K)]$. But, when $b \leq f$, low-priority requests are ignored.

The motivation of defining value k randomly is not making the key-demand deterministically depend on the priority to make weaponizing this dynamics harder for the adversary.

Additionally, to examine application behavior and separate legitimate from malicious queries, KMS determines the value of the supported `TTL` parameter as follows:

- KMS keeps a `session list` which contains `KSID` session identifiers that were served in the previous period. This list can also be stored on the SDN controller for centralized coordination.
- During a new `OPEN_CONNECT` query, it is checked whether the query contains the `KSID` identifier as defined in [10]. If there is no `KSID` identifier, KMS generates a new `KSID` for the session and stores it in the `session list`. Then the value of the `TTL` parameter is set to the initial value. The SDN controller can manage this value.
- If the `KSID` value already exists in the `session list`, then the `TTL` value will be incremented. The intention is to support those legitimate applications that follow the instructions received from the KMS and do not make additional `OPEN_CONNECT` requests.
- If KMS recognizes in the firewall query filtering algorithm [13] that the application generates new `OPEN_CONNECT` queries with the same `KSID` value (or new `KSID` value from the same IP address) and the previously reserved keys for that `KSID` session have not yet been served, then the `KSID` value is removed from the `session list` to sanction such aggressive behavior.

To summarize, low-priority queries can be processed only when $b > f$. However, an attacker can generate queries of higher priority. To prevent such malicious requests, in situations when $b < f$, KMS considers the `session list` and serves only those applications that have already been evaluated as reliable (those already in the `session list`). It is assumed that post-processing and other critical signaling/routing applications require keys constantly, and therefore they should be enabled for smooth continuous operation.

C. IMPLEMENTATION

To implement this formal mechanism, it is useful to review the concept of a Key Association Link as specified by ETSI 015. This is a logical relation established between two remote software-defined (SD) QKD nodes, which may or may not share a physical quantum connection. It is a virtual connection that appears as its own application to the KMS, and reserves keys for the intended lot of data to be transported. To this end, ETSI 015 specifies a set of parameters, among which the following are relevant for our simulation (from [17]):

- `bandwidth` (`uint32`): Required bandwidth (in bits per second) for that key association link. Used to reserve bandwidth from the physical QKD links to support the virtual key association link as an internal application. This is the variable N in (3).
- `Performance/eskr` (`uint32`): Effective secret key rate (in bits per second) generation of the key association link available after internal consumption. This is the variable λ in (3).
- `Performance/expected_consumption` (`uint32`): Sum of all the application's bandwidth (in bits per second) that are on this particular key association link. Depending on the choice of k, this is exactly the expression on the left side of (2), and thus controllable by the KMS.

For the requesting application(s), the SD-QKD node can then set the parameters for the QKD application based on the above choices, which are, following ETSI 015:

- `QoS/max_bandwidth` (`uint32`): regarding the requested quality of service, the maximum bandwidth (in bits per second) allowed for this specific application is equal to N, if (3) is satisfiable for some $k \geq 0$, and λ, i.e., the parameter "Performance/eskr" otherwise, if we designate all regenerated key material to the requesting application.
- `QoS/TTL` (`uint32`): This parameter corresponds to the maximum lifetime (in seconds) of the key reserved for given application without being used. If the key is not used in the specified time and there are no other keys reserved for the application that could be served, it means that the session has expired and it is necessary to create a new session using a new `OPEN_CONNECT` request. The SDN controller and can adjust parameters `max_bandwidth` and `TTL` to tune the number of sessions supported.

V. SIMULATION SETUP

The simulations were performed using the QKD Network Simulation Model (QKDNetSim[6]) [8] with installed ETSI 004 applications. The network topology consists of two

[6] The web version is available on www.open-qkd.eu

KMSs and one SDN controller that monitors and manages the associated QKD links using ETSI 015 based communication. To focus on the end-user experience of accessing key, key-relay functionality was not activated. Thus, we simulated multiple point-to-point links that connect nodes under local KMS entities' control. This implies that no routing protocols are used since directly connected nodes are simulated. A malicious application has been implemented that requires keys with no `KSID` values included in the `OPEN_CONNECT` query.

Simulations included static (no mobility) KMS nodes which were randomly placed in a rectangular region and connected to a single QKD link with the following settings: minimal amount of key material 10 kBits; maximal amount 100 kBits; buffers were initially empty; charging key rate was randomly selected from the [5000, 10000, 15000] bps while the size of keys was randomly selected from the range [1024, 2048, 4196, 8192] bits. The motivation for using a single QKD link is to reduce the number of available keys, i.e., analyzing the behavior of KMS in cases of congestion.

Simulations included up to 1, 5, 10, and 15 fully-operational ETSI 004 applications connected to KMS. Each application randomly selected OTP encryption or AES 256 with lifetime values of [10k, 20k, 100k, 200k, 300k, 400k, 500k] bytes. Also, each application randomly selected authentication type [unauthenticated, VMAC, SHA2]. Applications generated UDP traffic with a traffic rate randomly selected in the range of [1, 5, 10, 20, 30] kbps and fixed packet size randomly selected from the range [100, 300, 500, 800 1100] bytes. Also, applications randomly selected size of buffers to store encryption keys from the range [1, 3, 5, 10, 15, 20] and authentication keys from the range [6, 10, 15, 20, 50]; randomly selected hold-time interval from the range [0.5, 1, 3, 5] seconds. Each simulation was repeated using ten different random seeds, which ensured the randomness of differently selected values. The duration of the simulation was 150 seconds. The parameters not given here are the default parameters of QKDNetSim.

The malicious applications are installed to simulate a DoS attack against the KMS entity. Malicious applications are simplified end-user applications that do not perform any processing of received responses. Their goal is to send invalid `OPEN_CONNECT` requests to KMS at regular intervals. To model the intensity of the DoS attack, the experiment is carried out with a varying number of malicious applications. Also, the parameter p is defined, which determines the time interval between sending consecutive queries to the KMS with randomly defined values. The parameter p takes values from the set 0.05, 0.1, 0.3 and 0.5 seconds. To analyze the behavior of the `session list` mechanism, all application queries were low priority. In total, 3600 simulations were performed.

VI. SIMULATION RESULTS

If the attacker sends randomly generated `KSID` in the `OPEN_CONNECT` request, KMS will detect that `KSID` value has not been agreed with the peer KMS, and it will discard the request. However, in simulated scenarios, the firewall option on KMS was disabled, which means that queries from the same IP address were not automatically rejected, and the processing algorithm based on the `session list` was not activated for malicious applications. The attacker was sending `OPEN_CONNECT` without defining KSID value. Therefore, KMS processed requests and assigned `QoS/max_bandwidth` according to the priority and default `QoS/TTL`. Figure 3 shows the total number of generated queries to KMS from legitimate and malicious applications. It shows that the number of requested chunk keys increases exponentially with the number of malicious applications and the intensity of the attack.

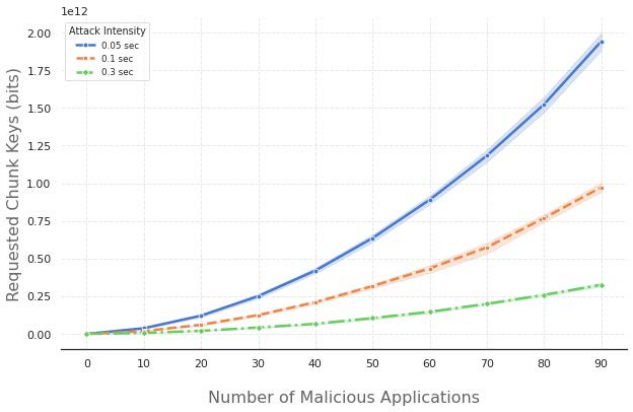

FIGURE 3. Amount of requested keys (the number of queries multiplied by the specified `chunk_size` value in the query) vs the number of malicious applications and their attack intensity.

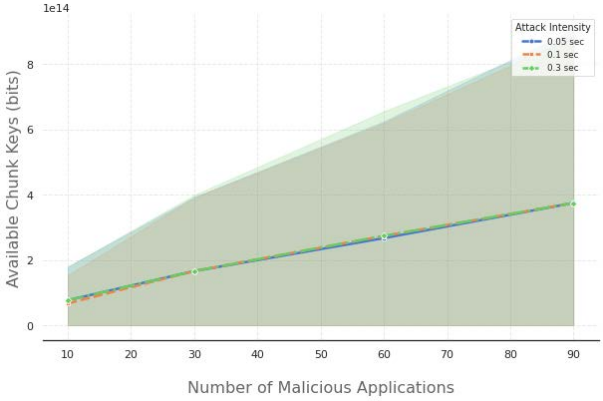

FIGURE 4. Amount of available keys (the eskr value divided by the defined `chunk_size` value in the query) vs the number of malicious applications and their attack intensity.

However, KMS can balance the increased intensity of attacks thanks to the algorithm for randomly selecting the value of k, which is based on the priority and the amount of the available key. Figure 4 shows the number of available keys depending on the intensity of the attack and the malicious applications. It can be seen that KMS is scarcer in the number of allocated resources in cases where a stronger attack is detected.

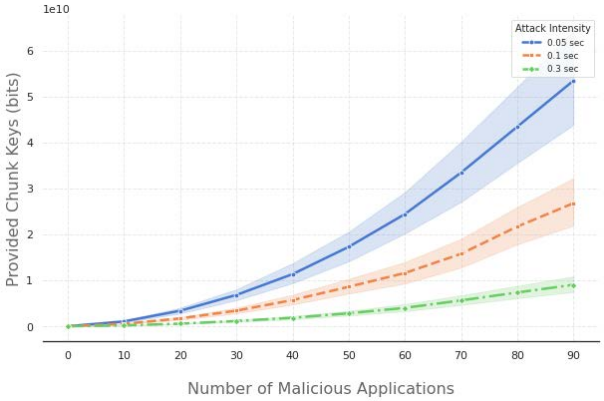

FIGURE 5. Amount of provided (allocated) keys vs the number of malicious applications and their attack intensity.

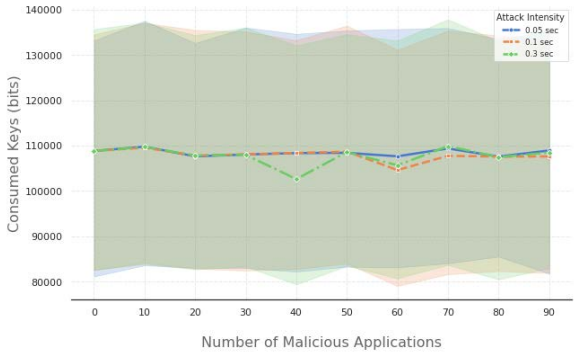

FIGURE 6. Amount of consumed keys by legitimate applications vs the number of malicious applications and their attack intensity.

Given that KMS cannot detect which applications are malicious and which are legitimate without the support of a firewall system, it will assign the appropriate amount of keys to all queries. Figure 5 shows the amounts of allocated keys in relation to attack strengths and the number of malicious applications. It can be noticed that the curves of the provided keys follow the shape of the curves of the requested keys. But, the amount of provided keys is significantly smaller than the requested ones, and the variability of the data is considerably higher (lightly shaded colors on the line diagram).

The success of the KMS algorithm can be further analyzed through the aspect of overall key consumption. From Figure 6, it can be observed that the amount of keys consumed by legitimate applications is almost constant. Legitimate applications are not embedded in DDoS attacks, which confirms the dynamic operation of selecting the number of served keys (parameter k) depending on the number of available keys and the priority of the applications.

Additionally, it is interesting to note Figure 7 which shows the averaged summed assigned `TTL` value. Given that malicious applications generated `OPEN_CONNECT` queries without a `KSID` value, which without the support of firewall mode KMS cannot classify as malicious traffic, each query was answered with the default value of `TTL`. That is why the curve follows a linear growth (`TTL` values do not depend on the `chunk_size` parameter, which is variable), regardless of the intensity of the attack.

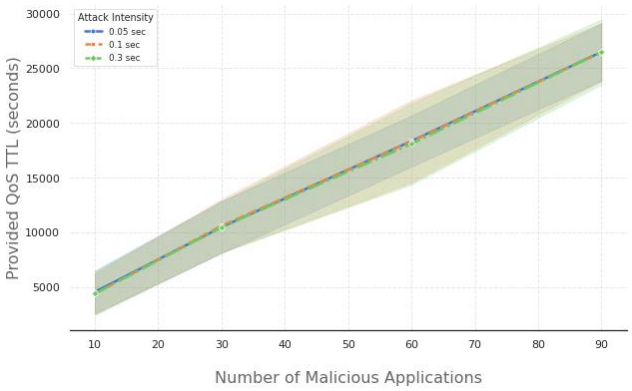

FIGURE 7. Average summed amount of assigned TTL values in relation to the number of malicious applications and traffic intensity.

VII. CONCLUSION

With the trend of incorporating QKD technology into everyday telecommunication networks, it is critical to be aware of security issues that might result in suspending functional operations. KMS must consider the number of requests issued by SAE apps and the number of cryptographic keys provided to SAE applications. If an attacker obtains such terminal devices, he can launch attacks to disrupt network entities' functional operations. Therefore, KMS should be able to identify malicious applications and be resistant to their attacks. The network should not be brought to a situation with insufficient keys, which would mean the cessation of functional work.

This paper presents a novel approach for tackling DDoS attacks in QKD networks. We have presented algorithms by which KMS can determine the number of keys assigned to applications and other supporting parameters such as TTL. Since KMS cannot fully trust the intended applications, it often avoids meeting all requested requirements. However, to avoid jeopardizing the security of the applications, we proposed a new approach of linear transformation that enables more secure data transmission in situations with scarce amounts of keys. Simulation experiments confirmed the credibility of the proposed solutions.

REFERENCES

[1] G. Brassard and L. Salvail, "Secret-key reconciliation by public discussion," in *Proc. EUROCRYPT*, vol. 765, 1994, pp. 410–423.
[2] M. Mehic, P. Fazio, S. Rass, O. Maurhart, M. Peev, A. Poppe, J. Rozhon, M. Niemiec, and M. Voznak, "A novel approach to quality-of-service provisioning in trusted relay quantum key distribution networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 168–181, Feb. 2020.
[3] N. Hosseinidehaj, Z. Babar, R. Malaney, S. X. Ng, and L. Hanzo, "Satellite-based continuous-variable quantum communications: State-of-the-art and a predictive outlook," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 881–919, Jun. 2019.
[4] M. Mehic, O. Maurhart, S. Rass, D. Komosny, F. Rezac, and M. Voznak, "Analysis of the public channel of quantum key distribution link," *IEEE J. Quantum Electron.*, vol. 53, no. 5, pp. 1–8, Oct. 2017.
[5] M. Mehic, M. Niemiec, S. Rass, J. Ma, M. Peev, A. Aguado, V. Martin, S. Schauer, A. Poppe, C. Pacher, and M. Voznak, "Quantum key distribution," *ACM Comput. Surveys*, vol. 53, no. 5, pp. 1–41, Oct. 2020.
[6] E. Dervisevic and M. Mehic, "Overview of quantum key distribution technique within IPsec architecture," in *Proc. 18th Int. Conf. Inf. Syst. Crisis Response Manag. (ISCRAM)*, 2021, pp. 1–10.

[7] S. König and S. Rass, "On the transmission capacity of quantum networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 2, no. 11, pp. 9–16, 2011.
[8] M. Mehic, O. Maurhart, S. Rass, and M. Voznak, "Implementation of quantum key distribution network simulation module in the network simulator NS-3," *Quantum Inf. Process.*, vol. 16, no. 10, p. 253, Oct. 2017.
[9] V. Martin, J. P. Brito, C. Escribano, M. Menchetti, C. White, A. Lord, F. Wissel, M. Gunkel, P. Gavignet, N. Genay, O. Le Moult, C. Abellán, A. Manzalini, A. Pastor-Perales, V. López, and D. López, "Quantum technologies in the telecommunications industry," *EPJ Quantum Technol.*, vol. 8, no. 1, p. 19, Dec. 2021.
[10] *ETSI ISG QKD 004—Quantum Key Distribution (QKD); Application Interface*, Standard ETSI GS QKD 004 V2.1.1, Apr. 2020, pp. 1–22.
[11] S. Rass and K. Sandra, "Indirect eavesdropping in quantum networks," in *Proc. 5th Int. Conf. Quantum, Nano Micro Technol. (ICQNM)*, 2011, pp. 83–88.
[12] S. Rass and S. König, "Turning quantum cryptography against itself: How to avoid indirect eavesdropping in quantum networks by passive and active adversaries," *Int. J. Adv. Syst. Meas.*, vol. 5, no. 1, pp. 22–33, Dec. 2012.
[13] E. Dervisevic, F. Lauterbach, P. Burdiak, J. Rozhon, M. Slivova, M. Plakalovic, M. Hamza, P. Fazio, M. Voznak, and M. Mehic, "Simulations of denial of service attacks in quantum key distribution networks," in *Proc. 28th Int. Conf. Inf., Commun. Autom. Technol. (ICAT)*, Jun. 2022, pp. 1–5.
[14] P. Schartner and S. Rass, "Quantum key distribution and denial-of-service: Using strengthened classical cryptography as a fallback option," in *Proc. Int. Comput. Symp. (ICS)*, Dec. 2010, pp. 131–136.
[15] S. Rass and P. Schartner, "Information-leakage in hybrid randomized protocols," in *Proc. Int. Conf. Secur. Cryptogr. (SECRYPT)*, J. Lopez and P. Samarati, Eds. Setúbal, Portugal: SciTePress, 2011, pp. 134–143.
[16] G. Goos, J. Hartmanis, J. van Leeuwen, and V. Boyko, "On the security properties of OAEP as an all-or-nothing transform," in *Proc. Adv. Cryptol. (CRYPTO)*, vol. 1666, M. Wiener, Ed. Berlin, Germany: Springer, 1999, pp. 503–518.
[17] *Quantum Key Distribution (QKD); Control Interface for Software Defined Networks*, Standard Standard ETSI GS QKD 015, V2.1.1, Apr. 2022.

MIRALEM MEHIC (Member, IEEE) received the Ph.D. degree in telecommunications from the VSB–Technical University of Ostrava (Czechia). He also studied from the AGH University of Science and Technology, Krakow, Poland; Alpen-Adria-Universitat Klagenfurt, Austria; and the Austrian Institute of Technology (AIT) in the Department of Digital Safety and Security Business Units–Optical Quantum Technology, Vienna and Klagenfurt, Austria. Since 2019, he is the Head at the Department of Telecommunications, University of Sarajevo, Bosnia and Herzegovina. He is the author of the *Unique QKD Network Simulator QKDNETSIM*. His research interests include quality of service and management of QKD networks with a focus on real-time traffic and the utilization of network resources. He is the national principal investigator of EU H2020 OPENQKD and the NATO SPS G5894 QUANTUM5 projects (For more details: www.qkdnetsim.info).

STEFAN RASS received the degree in mathematics and computer science. He is a Full Professor at the Johannes Kepler University in Linz, and an Associate Professor at Universitaet Klagenfurt. He authored numerous papers and books related to security risk management, security infrastructures, applied quantum cryptography, applied statistics, and decision theory in security. He is involved in various nationally and internationally funded research projects, and being a contributing researcher in many EU projects and offering consultancy services to the industry. His research interests include decision theory and game-theory with applications in system security, especially robotics security, and complexity theory, statistics, information-theoretic security, and applied cryptography.

EMIR DERVISEVIC was born in Berlin, Germany, in 1995. He is currently pursuing the Ph.D. degree with the Department of Telecommunications, Faculty of Electrical Engineering, University of Sarajevo. Since 2020, he has been a part of the international scientific research projects Horizont 2020–Open European Quantum Key Distribution Testbed (OPENQKD) and NATO SPS MYP G5894–Quantum Cybersecurity in 5G Networks (QUANTUM5). He is actively developing the quantum key distribution network simulation module. His research interests include quantum key distribution networks, network management, network security, and cryptography (For more details: www.open-qkd.eu).

MIROSLAV VOZNAK (Senior Member, IEEE) received the Ph.D. degree in telecommunications from the Faculty of Electrical Engineering and Computer Science, VSB–Technical University of Ostrava, in 2002, and the Habilitation degree, in 2009. In 2017, he was appointed as a full professor of electronics and communications technologies. He has authored and coauthored over 100 articles indexed in SCI/SCIE journals. His research interests include information and communication technologies, quality of service and experience, network security, wireless networks, and big data analytics. According to the Stanford University study released in 2020, he is one of the World's Top 2% of Scientists in Networking and Telecommunications and Information and Communications Technologies.