Key management and control systems for QKD

by Jasmin Neumann, Felix Trunk, Susanne Naegele-Jackson

Content

Introduction

Quantum Key Distribution (QKD) is a method for generating secure cryptographic keys between network participants. So far, some QKD systems have been well researched and are commercially available. But the mere generation of the keys is only the first step for a cryptographic application; further steps are key management and key distribution: Generated keys must be able to be stored securely and systematically in a key buffer with an appropriate buffer size in order to be available in time if necessary. Security must always be maintained in the form of authentication and key lifecycle management. Key Management Systems (KMSs) must also ensure that the keys are distributed via suitable links in the network in such a way that they are available at the target application at the right time. For this purpose, there are control structures similar to a classic network that support key management. In addition, the combination with Software Defined Networking (SDN) is also suitable.

In the following, the KMS and its tasks are presented in detail; the control plane in a QKD Network (QKDN) is described in more detail as well. Subsequently, the open questions regarding the current state of standardization (both at ETSI and ITU) at the various interfaces of QKD, KMS & control systems and their required key monitoring parameters are discussed.

1. The Key Management System (KMS) in the QKDN

In a network for quantum key distribution, a distinction is made between several layers (see ***Figure 1***): At the lowest level is the physical layer (Quantum Forwarding Plane (QFP); blue in the image) in which the quantum keys are generated; the key management layer (green) above is responsible for managing and forwarding the keys. Above lies the QKDN Control Layer (orange; explained in the following Chapter ***Control systems in the (QKD) network*** in more detail). The top layer is the application layer (gray) where the keys are consumed by the applications.

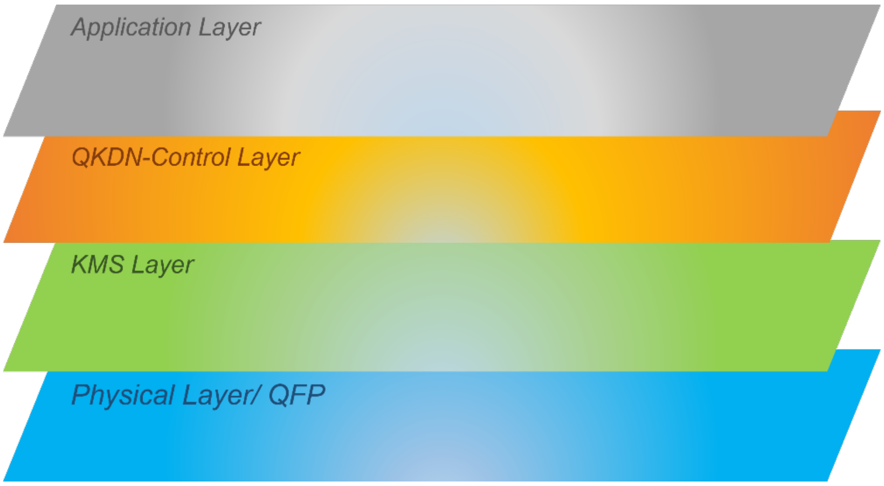

Figure 1: *Model with hierarchical layers of QKDN*

On the one hand, a KMS of the KMS layer receives the generated keys of the QKD devices (Physical Layer/QFP) and stores them in various key buffers. The KMS can not only request the keys from the lowest layer, but is also able to synchronize with the other KMSs at other nodes for key distribution. On the other hand, the KMS ensures that the key pairs generated securely by QKD are distributed to the applications over the routes according to specified routing and are also available in the correct format when the application requires them.

The basic functions of a KMS include:

- Secure storage of keys in buffers, including formatting of the keys (Function 1)
- the stocking/clearing of common key storage, as well as key exchange (relaying) based on predefined routes (Function 2)
- Timely delivery of keys to the applications (supply) according to specified Quality of Service (QoS) (Function 3)

Other tasks of a KMS include:

- synchronization, authentication, and management of key databases
- the validation of the fill levels of these buffers, as well as the associated timely new request at an empirically determined threshold value
- Participation in the key rotation (replacing the old keys so that the data encrypted by the current key is limited in case it does become public).

The ETSI Standard ***ETSI GS QKD 004*** distinguishes between a QKD Key Manager Peer (South-Bound KMS) per QKD device, which receives the keys at the QKD Link Level, and a QKD Key Server (North-Bound KMS) per node at the QKD Network Level (see ***Figure 2***).

The key servers (Functions 2/3) are also needed to enable the exchange across several QKD devices. The key servers use the same application interface for key transport with the key manager peers (green arrows) as well as with the applications (orange arrows).

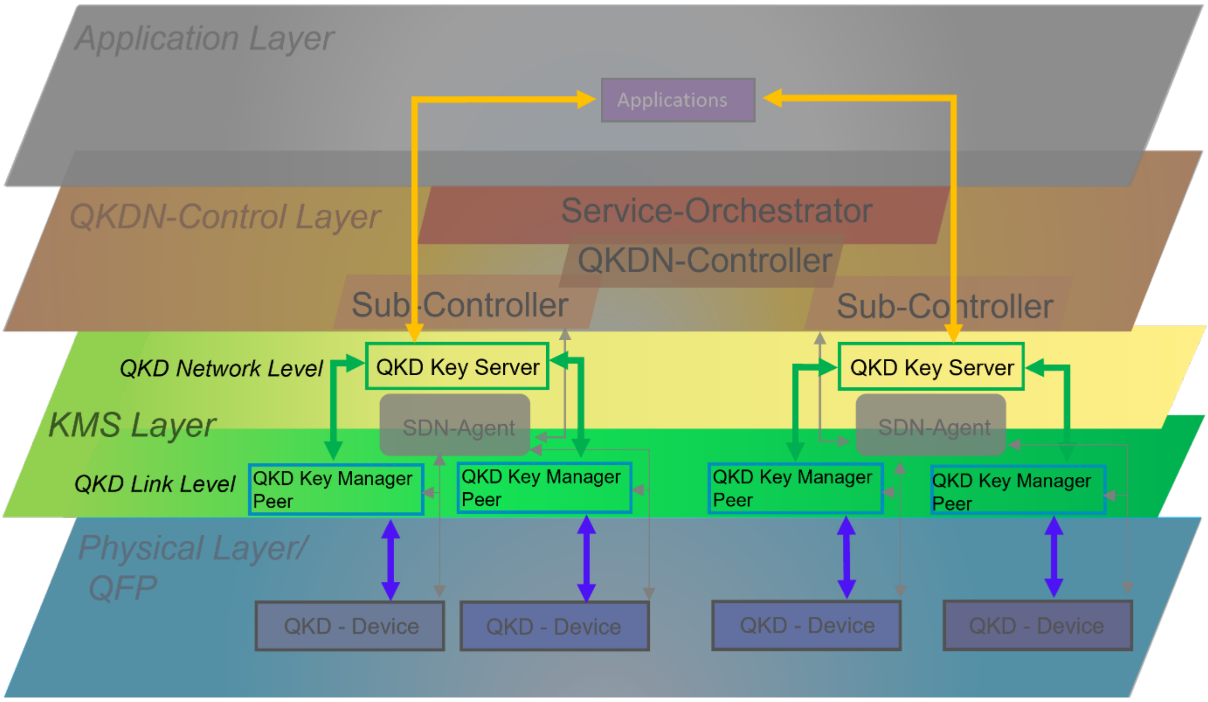

Figure 2: *KMS hierarchy according to ETSI*

(Thick colored arrows indicate the vertical key transport)

The ITU-T Recommendation ***ITU-T Y.3803*** also divides the KMS into two levels: quantum-oriented key management agent (KMA) and application-oriented key supply agent (KSA) (see ***Figure 3***).

The KMA takes care of the key relay, i.e., the transport between KMSs of different trusted nodes (the concept will be explained in more detail below), the storage and the key life cycle management, while the KSA is familiar with the provision of the keys for the application (and with the combination of other keys). Instead of the SDN agent that belongs to the control layer in ETSI, there is also a special control and management module directly in the KMS, QKD & Control layer. These management modules are not only linked to each other (via the interfaces *Ck, Kq-2*), but also to a separate, all-encompassing management layer (e.g. via interface *Mk*). One can also see here how the keys from the QKD layer to the KMS layer are exchanged via the *Kq-1* interface and from there via the *Ak* interface directly with the application, while a control layer is also interposed here. The *Kx* interfaces are intended to enable a key exchange between KMSs.

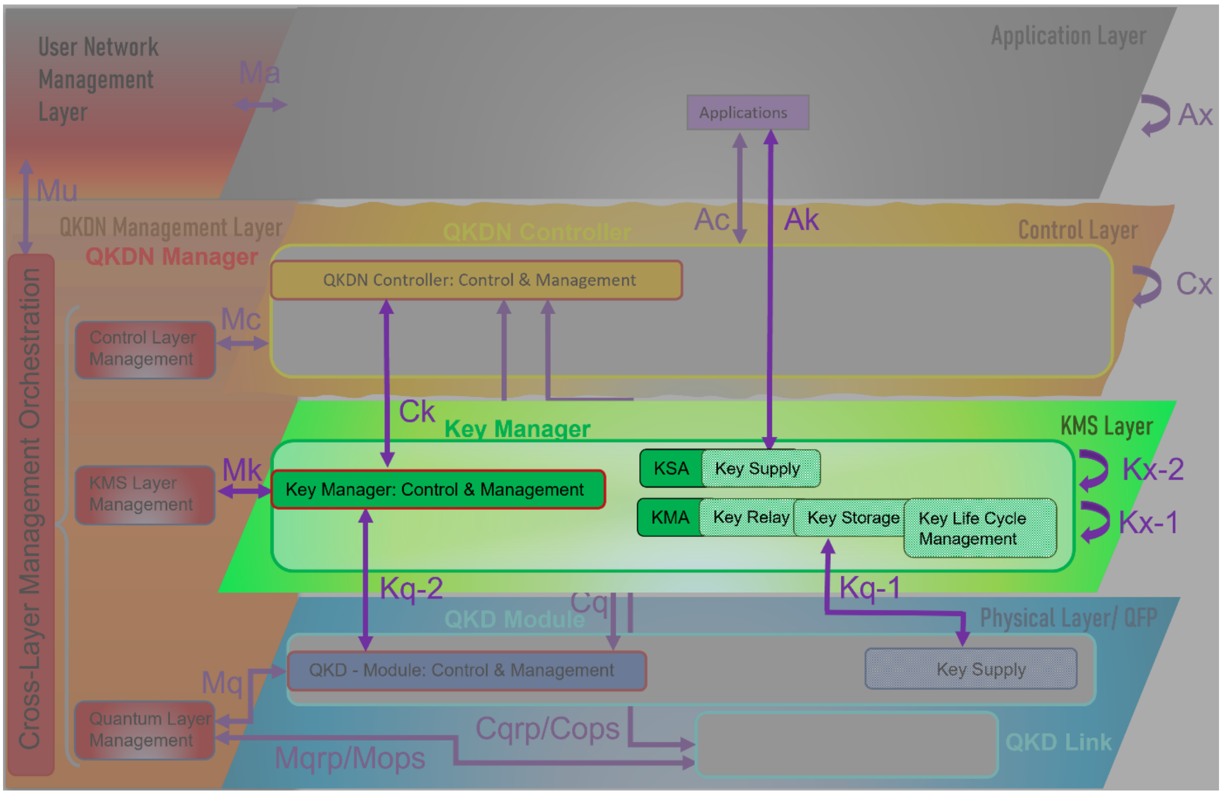

Figure 3: KMS hierarchy according to ITU-T

More detailed explanations of ETSI and ITU-T standards can be found in the Chapter ***Interoperability & Standardization.***

According to [1] when issuing keys to the application, a distinction is made between systems with one-time-keys and key-streams. The former is the key request for each individual operation with a new, independent request via a key-ID. The advantage here is that there is no need to cache any states about previous key requests. However, this also means that it is not possible to maintain automatic QoS (e.g., minimum throughput of keys, network paths) because there is no way to make any predictions about key consumption. In addition, there is a certain time delay due to the renewed requests. In contrast, key-streams come with a session concept. For this purpose, a key-stream is defined between the two applications for simultaneous key distribution via a key-stream-ID, which allows the keys to be transmitted sequentially. Here, a QoS can be inherently implemented, which can also be easily extended to new QoS aspects. This method is particularly suitable for real-time applications with different quality requirements, such as those found in streaming. For this purpose, the key-streams according to ETSI GS QKD 004 are also much more complex and require more resources. On the other hand, one-time-keys without QoS, can be implemented comparatively easily using RESTful APIs (e.g. ETSI GS QKD 014).

However, current developments show that it makes sense to combine the two standards accordingly for development in a real environment. It is often referred to as "ETSI 004 +". The aim is to combine the rich parameter selection (QoS) of ETSI GS QKD 004 with a simple implementation.

The process of the key request from the application to the KMS must meet certain criteria [1]:

1. Key generation: In order to be able to request the QKD devices along the predefined path to produce the key material between the paired QKD devices, the QKDN-controller must interact with the local KMSs along the path accordingly and specify concrete path information for planning the key requirement. Thus, as can already be seen in the layer model, the KMS occupies an intermediary position between the QKD layer and the control layer (cf. ***Figure 1***).

2. Key classification: Keys have different properties that are important for the user or the application (e.g. the length) and different network parameters (e.g. the hops via trusted nodes). The different local KMSs have to work together to divide them into classes. When distributing keys, an unambiguous identification must also be ensured.

3. Key distribution: After classification, the keys can be routed accordingly.

4. Key storage: In order not to have to generate keys instantaneously when an application is requesting keys, the KMSs use buffers to absorb peaks in demand and to bridge the time gap of key creation. This storage must be standardized, secure and carefully planned in the KMSs.

5. Key assignment: In cases without initial key request of an application where the buffers were filled with keys for general availability, there must be protocols in place that divide the existing keys fairly in the case of several simultaneous requests, and if necessary, also taking QoS considerations into account.

Whenever there is no direct connection via a QKD link of two paired QKD devices, the so-called key relaying takes place, in order to enable a key exchange between network participants. The route is bridged via individual point-to-point (P2P) connections between the nodes with individual, direct keys, so that the entire route is connected via an indirect key (see ***Figure 4***) [1]. This can be created, for example, by XOR operations between the direct keys [2]. There is also the possibility of key relay by means of encrypted random numbers. For security reasons, however, it is preferable that the encrypted keys are sent directly to the target node/central node and are only decrypted there.

In [3] for example, an efficient optimization for a key relay algorithm was developed to keep the required QKD keys as low as possible when transitioning between trusted nodes. Nodes are only selected for the route if this results in the lowest possible requirement for keys.

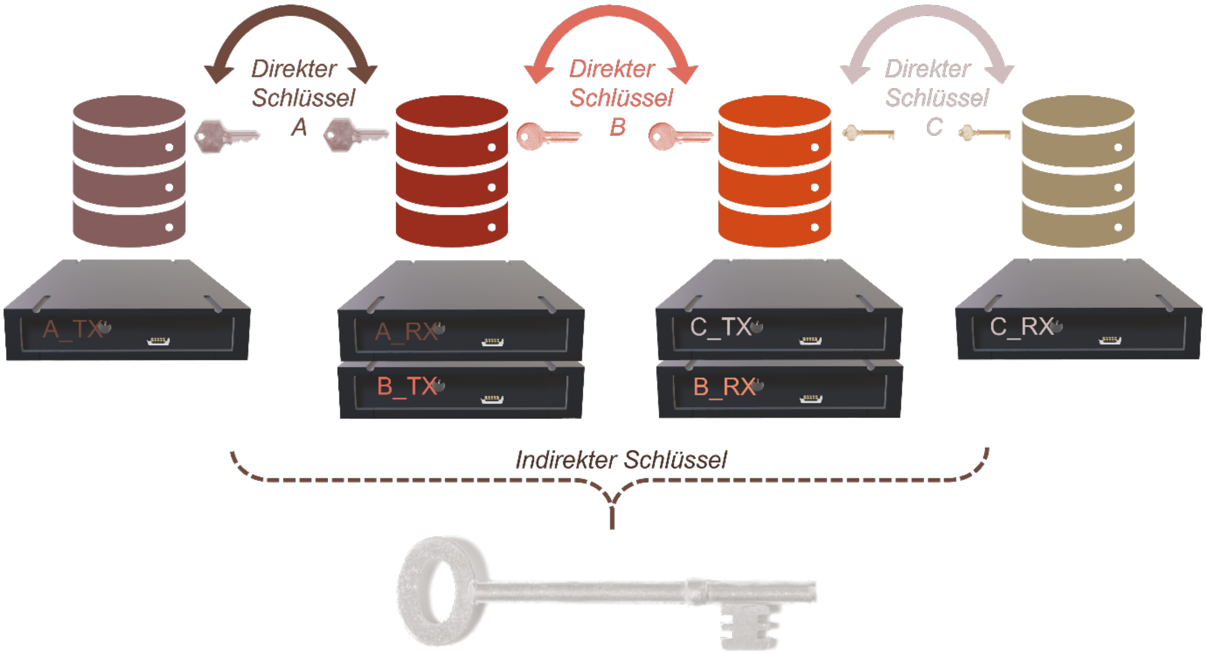

Figure 4: *Key exchange via four trusted nodes according to* [1]

Routing is not one of the tasks of the KMS [NIST SP 800, [4]], but the QKDN-controller calculates the optimal route. The routing of the keys can be controlled centrally or distributed with QKDN-controllers and is generally linked to SDN. However, the controller does not transport the keys itself, but the KMS supplies the cryptographic application with keys via the route calculated by the QKDN controller [5]. More information about controllers in the QKDN can be found in Chapter ***Control systems in the (QKD) network.***

Finally, the indirect keys are dispensed to the application so that it can encrypt the user data accordingly. For this the location of the storage of the keys by the corresponding applications must be determined. A distinction is made between peer-to-peer (key material is stored at the consumption node/client computer) and service-to-service (key material is stored on an application server). In the latter case, the key and the encrypted message are stored with corresponding user data in order to have it ready for the request. Service-to-service includes another security-critical aspect: the data between the client computers and the application servers is transmitted in plain text [1]. In order to maintain security, these links must be encrypted with Post Quantum Cryptography (PQC).

In addition, there is a key lifecycle management that, depending on the global policy, deletes the unused keys in the KMS buffers after a certain period of time after a new request [6]. For this purpose, time constraints similar to the key-ID must be carried along when the key is transmitted. The key rotation of the old keys used in the application, as well as the synchronization, also count as part of the lifecycle management of the keys.

Of course, the security aspect is of particular importance for KMSs, as the key exchange between KMSs must be protected against potential attackers during the transport over the network and also especially at the trusted nodes. Securing the KMS includes authentication, authorization and monitoring. In a KMS attack, the KMS itself is attacked so that it cannot supply the encryptors with keys [7]. KMSs offer the weaknesses that they cannot handle many requests from the applications at the same time, because the key buffers can run out if not enough key material can be produced quickly enough [8]. In general, a failure scenario must therefore be defined, whereby PQC is also a suitable substitute here.

However, although every QKDN must have a corresponding key management system and SDN is also being used more and more frequently, this central and important point in the QKDN often raises open questions about the implementation of the real interaction between the different layers, especially for devices from different manufacturers. In order to enable smooth communication between all network components, well-defined interfaces are required in particular, which are described in Chapter ***Interoperability & Standardization.***

2. Control systems in the (QKD) network

A modern network based on automatic elements and software-defined networking (SDN) uses SDN for control, i.e. with the help of software, the data plane is separated from the control plane and higher-level SDN instances are used, which maintain the central logic of control in the network. In addition, the processing of a service is monitored so that the customer's service request can be implemented correctly with the necessary resources. In the following, the meaning of SDN/QKDN-controller, network orchestration and management is clearly defined and differences in the terms regarding ETSI and ITU-T are pointed out.

A classic network controller is responsible for controlling the network components and abstracts access to the associated resources of the (classic) network for the service-orchestrator, who is responsible for implementing the entire service. If a network also offers quantum key distribution, for the quantum domain a QKDN-controller is used. To combine both networks, ETSI uses an SDN-orchestrator (~ Service-Orchestrator), which can thus also grant the transition/routing between different domains, see ***Figure 5***. The SDN-orchestrator can automatically allocate the resources available in the network and performs similar tasks as an SDN-controller only across domains.

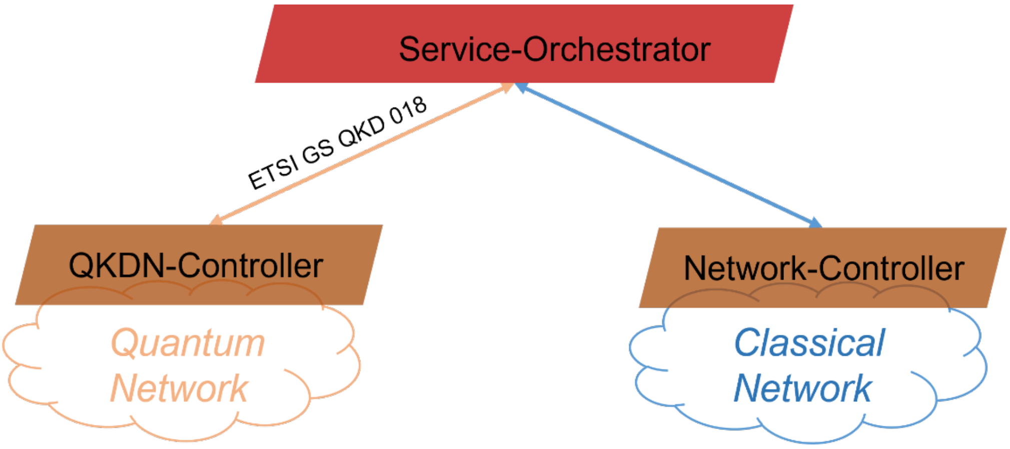

Figure 5: *Connection of QKDN-domain with classic domain in ETSI 018*

Controllers can be decentralized at the node as well as centralized. Depending on the network architecture, the QKDN-controller can also control multiple sub-controllers. During key exchange, the QKDN-controller calculates the route, but does not transport the keys itself;instead the KMS supplies the cryptographic application with keys, which is also the first to receive the service request. According to ITU-T, a QKDN-controller is responsible for the configuration of the QKD-network, routing, access control, as well as policy-based control and session control [9]. An SDN-controller virtualizes the QKDN and controls all of its programmable elements, and is also directly responsible for application registration and topology acquisition [10]. In addition, with ETSI, the controller abstracts the underlying part of the network it controls for the SDN-orchestrator, so that it can use the relevant information to optimize the network and allocate network resources.

According to ***ETSI GS QKD 015*** in the case of a centralized SDN-controller, additional SDN-agents are used at the node to control all devices located at the node (QKD-devices and KMS) according to the specifications of the SDN-controller. This includes both the configuration of the QKD modules with the abstracted information obtained, as well as the comparison of the states of the keys in the KMS buffers with the actual demand. In ETSI GS QKD 015, every SDN-capable QKD-node should also contain a KMS, which collects key material from different associations and can have many logical key stores. The applications with their QoS specifications should be registered by the KMS and the monitoring parameters should be provided.

Figure 6 shows the hierarchical control structures in a QKDN according to ETSI nomenclature, which has a central SDN-orchestrator, several SDN-controllers and several SDN-agents.

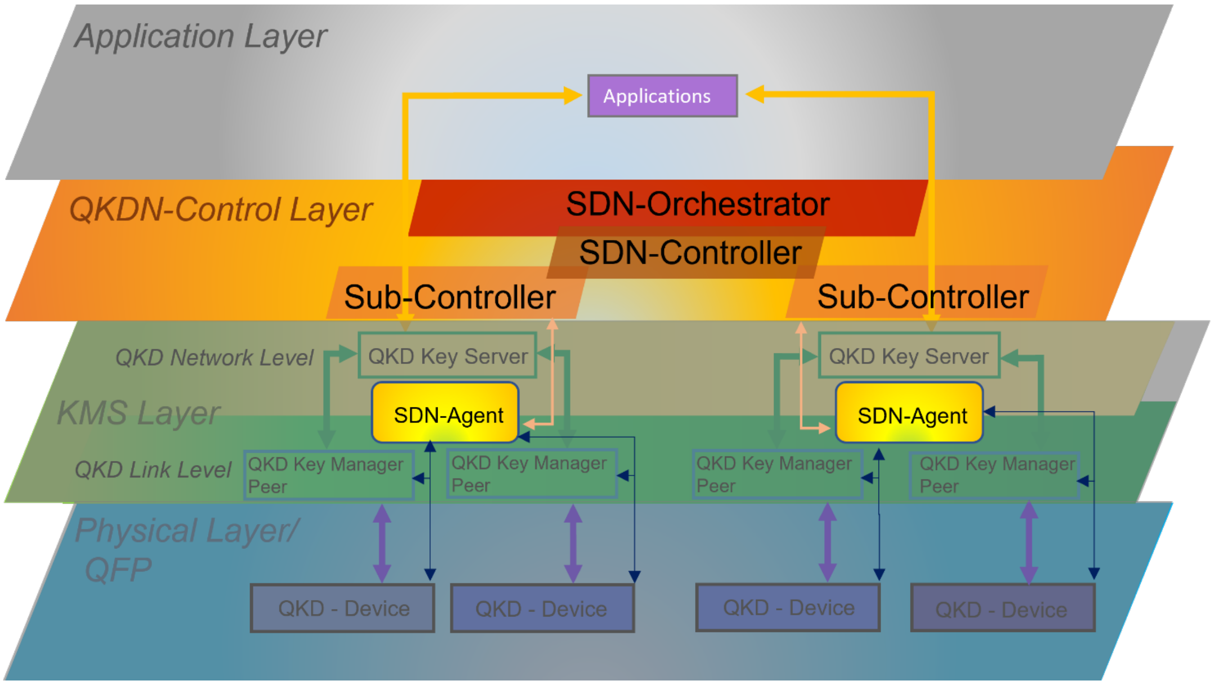

Figure 6: *Control structures of ETSI*

SDN-controllers/orchestrators also perform management tasks in parallel at ETSI, while at ITU-T there is a separate QKDN network manager that is specifically responsible for Fault / Configuration / Accounting / Performance / Security (FCAPS) management. The following management functions are particularly relevant for QKD key management: Managing the Key Supply Service Policy, tracing the key lifecycle management using a log database, or complying with the key management policies and forwarding them to the SDN-controllers.

The following ***Figure** 7* shows an overview of the control structures according to ITU-T. Although no hierarchy concept can be seen in the figure, several hierarchical levels are possible in which SDN-controllers link different sub-QKDNs with each other. The separate management layer, including cross-layer management, communicates with all layers via Control & Management submodules. The submodules can also communicate with each other via the interfaces *Ck/Cq/Kq-2*. Thus, they form an equivalent to the SDN-agents in the respective node of ETSI. Further details on the Network Manager can be found at ITU-T Y.3804 and in the subchapter ***Comparison with ITU-T*** of the following chapter.

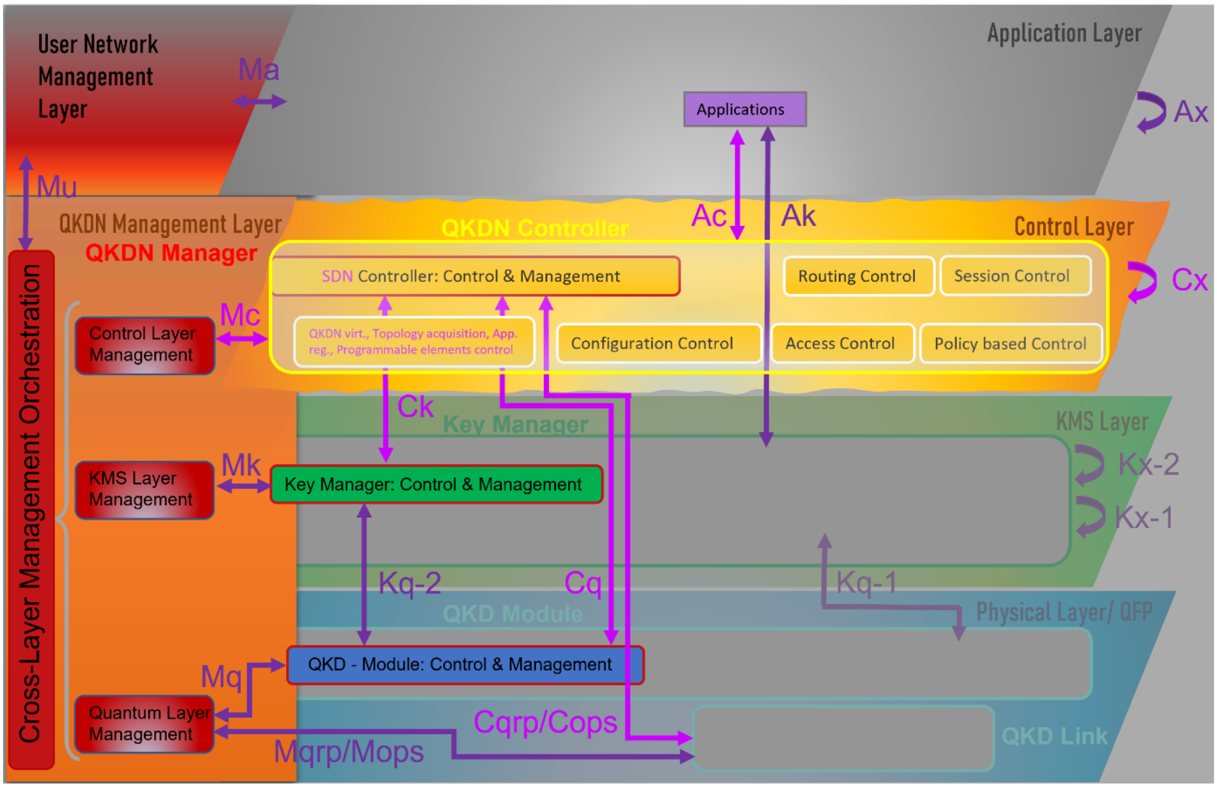

***Figure** 7: Control structures with SDN (pink) at ITU-T Y.3805*

In the following ***Table 1*** all control systems with their respective tasks are listed once again:

Key Management	Controller/Orchestrator	Network Manager
Key Formatting	Routing control between key relays	Error management
Extend key format with metadata (key-ID, date, length, etc.)	Control of the communication of the layers on request	Configuration management
Acquisition of QKD Link parameters (e.g. throughput)	Control QKD & KMS (-links)	Accounting Management
Storage in buffers	Configuration Control Reconfiguration in case of failure	Performance Management
Key relay between KMSs	Authentication/Authorization Access control	Security Management
Key Synchronization	Session control	Status Monitoring (QKD Module, QBER)
Key Lifecycle Management	Policy-based control	Supports KMS for key lifecycle management & supports controller for routing
Key delivery to application	FCAPS Management QoS/Charging	Management Authentication/Authorization
Key authentication using KMS links	**Controllers only:** Deploying Network Topology/Parameters Abstraction for Orchestrator	Management of QoS/Charging
	Orchestrator only: Coordination/optimization of SDN-controllers of different domains; Support routing across domains; Topology acquisition /service delivery across domains; Monitoring Control Parameters	

Table 1: *Overview of Network management systems and their responsibilities generalized according to ITU-T Y.3800, ETSI 004/014/015/018*

3. Interoperability & Standardization

In the case of KMS systems, interoperability is an important topic in order to be able to combine different QKD components with existing hardware in a network in practice.

However, especially in the area of network management, standardization is not yet as advanced as, for example, when handing over keys to the application.

Even though some efforts are already underway to increasingly standardize the interfaces, many are still being updated (e.g. ETSI GS QKD 014) or in draft. This also includes the standard that is intended to enable communication between different KMS manufacturers at a trusted node (ETSI GS QKD 020). Only when all ongoing standardizations have been completed, manufacturers will be able to design their devices accordingly and vendor-agnostic QKD networks will be increasingly easier to set up. Some of the major manufacturers (e.g., IDQ, Toshiba and ETSI) are also involved in the standardization itself.

An overview of the definition of various possible interfaces in a typical QKD network – including the corresponding existing ETSI standardizations and the planned ETSI GS QKD 020 – can be taken from ***Figure 8***:

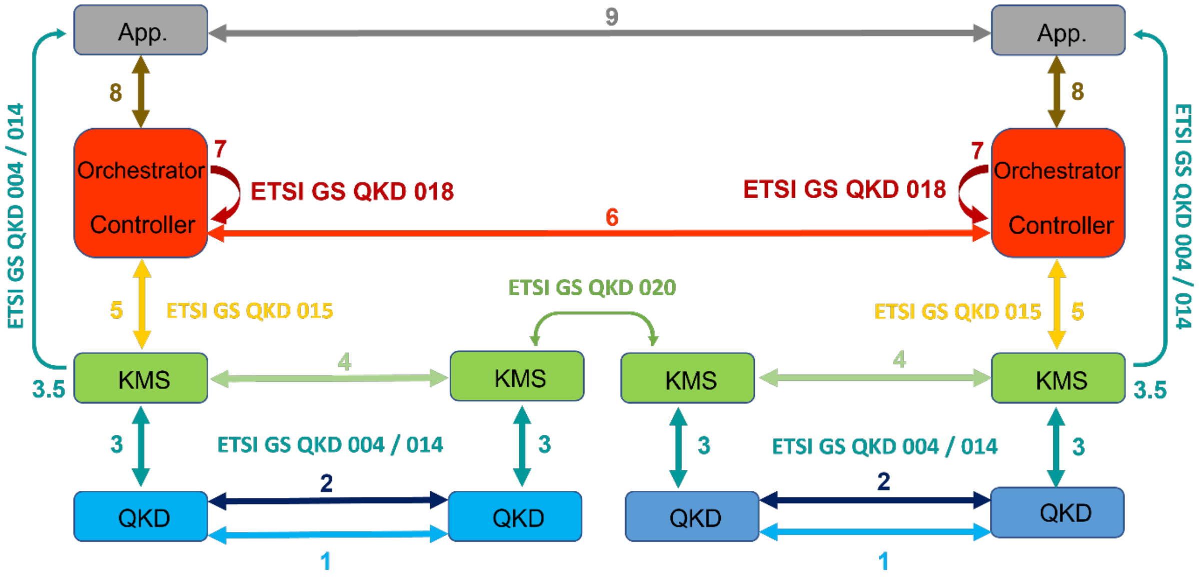

Figure 8: *Interfaces and ETSI standards according to* [1], [11]

There are both vertical interfaces (APIs between higher and lower instances) and horizontal interfaces between so-called peer-to-peer entities at the same level.

In ***Table 2*** the various interfaces and protocols of ***Figure 8*** are briefly introduced and explained [11]:

1	Quantum Level Communication Protocol: Transfer of QuBits to Quantum Channel
2	QKD Post Processing Protocol: Sifting, Error Estimation & Correction, Privacy Amplification on Classic Channel
3	QKD API for key transfer to the KMS: ETSI GS QKD 004/014, proprietary protocols (Cisco SKIP)
4	Exchange of key material between different KMSs from any manufacturer: proprietary protocols (Quantum Point-to-Point protocol from SECOQC [12]), ETSI GS QKD 020
5	Communication KMS with SDN-controller for connection establishment: ETSI GS QKD 015
6	Routing the keys between the different, non-adjacent KMSs
7	ETSI GS QKD 018 for the exchange between SDN-controller and SDN-orchestrator
8	API between SDN-orchestrator and application to establish/configure end-to-end application connections through the network (e.g. QoS)
3.5	API connection between KMS and application: ETSI GS QKD 004/014, proprietary protocols (Cisco SKIP)
9	Application-specific exchange of keys

Table 2: *Overview of SDN QKDN Interfaces*

ETSI Standards

The following is an overview of the most important standards for QKDN in relation to KMS and SDN (cf. [6]):

ETSI GS QKD 004 [13]

Here, the *push/pull based operation mode* both as a north-bound API between KMS and application is defined, and also as a south-bound interface that allows the KMS to talk to the QKD layer. This results in two vertical subgroups within the KMS: the QKD Key Manager Peer and a QKD Key Server (see Chapter ***The Key Management System (KMS) in the QKDN***). KMS enable horizontal exchange between transmit/receive nodes within the local security boundaries. Key requests with the parameter of a predefined QoS or the requests of a Key Stream-ID parameter from the KMS are possible. There should be a way to transfer metadata. In addition, every Key Stream has a Time To Live (TTL).

ETSI GS QKD 014 [14]

Similar to its predecessor, a comparatively simplified API is defined between KMS (here: Key Management Entity (KME)) and Application Layer (here: Secure Application Entity (SAE)), following the REST architecture principle. Communication is based on the HTTPS protocol with TLS 1.2 or higher. Here, only key-IDs are supported (no key-stream-IDs) (see Chapter ***The Key Management System (KMS) in the QKDN***).

ETSI GS QKD 015 [15]

This standard describes the interface between an SDN-controller and the KMS. The communication of the SDN-controller via an SDN-agent in the QKD node enables the combined control of the QKD module and the KMS by the SDN-controller (see Chapter ***Control systems in the (QKD) network***). The agent is supposed to provide information about registration and optimization to the controller, as well as information about the links to other QKD systems. The information for the controller should be here as abstract as possible. For this purpose, the YANG data modeling language is used. The components of the QKD node are therefore divided into four groups: The parameters for the QKD node, the QKD interfaces, the link to the QKD key assignment (direct/virtual) and the QKD application (external/internal). External applications are, for example, an end-user application and internal applications are authentication keys. The SDN controller also takes care of application management (registration/QoS, find peer nodes) so that the application does not have to manage this.

ETSI GS QKD 018 [16]

This is a standard specifically for orchestration with SDN. An SDN-orchestrator is used to control the network across the different domains controlled by SDN-controllers (see Chapter ***Control systems in the (QKD) network***). To do this, the general communication between the SDN-controller and the SDN-orchestrator must be defined. First, the topology of an SDN-controlled network with all its nodes is determined by the SDN-orchestrator, either proactively on each path recalculation or reactively following a change by the controllers. No information is collected about the applications that consume these keys, so service links still need to be determined. Monitoring parameters are then defined to describe the nodes, the links and thus the network status. This enables end-to-end service delivery by the SDN-orchestrator. In addition, the orchestrator can receive notifications (events, alarms) from the network components, including those sent to the SDN-controller. The interface always consists of the data model and the transport protocol. The former describes the language by which the data is exchanged, for which YANG will also be used here. The latter defines the communication rules via protocols such as NETCONF and RESTCONF. REST protocols are based on HTTP and, as a subset of NETCONF, support its functionality, which can read the YANG data.

ETSI GS QKD 020 (in draft) [17]

ETSI GS QKD 020 will ensure the interoperability of different KMSs from different manufacturers so that more vendor-agnostic devices can be used. It is scheduled to be published at the end of 2024. However, ETSI GS QKD 020 is only intended to apply to two KMSs within a node. Thus, a standardization of a KMS interface between arbitrary nodes is still missing.

Related to this, there will be another standard ETSI GS QKD 021, which enables the orchestration interface specifically for these interoperable KMSs in multi-domain QKDNs.

In the Spanish testbed MadQCI ([4], ***Figure 9***), however, it has been shown that this functionality can be easily realized by means of SDN-modules that have information about all network devices (including the key requirements of the KMSs): The use of SDN comes in conjunction with a local *SDN-agent*, which exists in addition to the *Local KMS (LKMS)* at the node and with which the entire rest of the node can be controlled. This means that end-to-end key transports can be implemented in a vendor-agnostic way between any nodes in the network. In addition, an extra *Forwarding Module* is used in the lowest layer for routing and the associated key transport. QoS parameters can thus be easily integrated into the routing. Therefore, the LKMS only acts as a managing key buffer that provides the SDN with information about the keys and then passes the keys on to the application.

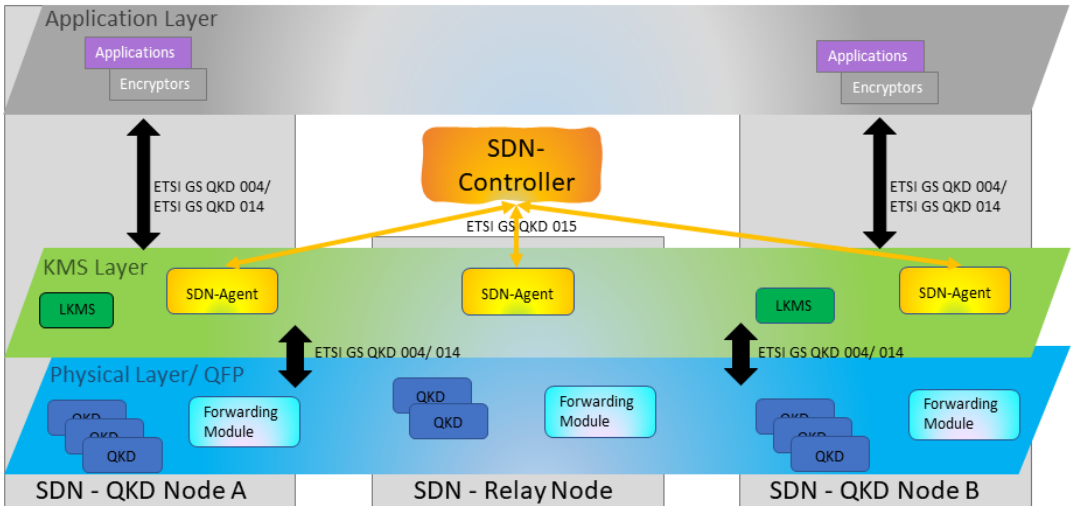

Figure 9: *Interfaces and Standards with SDN-Controller at MadQCI [4]*

The components of the KMS and Control layers of ETSI presented so far, lead to a general layer model as a result, as can be seen in a comprehensive view in ***Figure 10***:

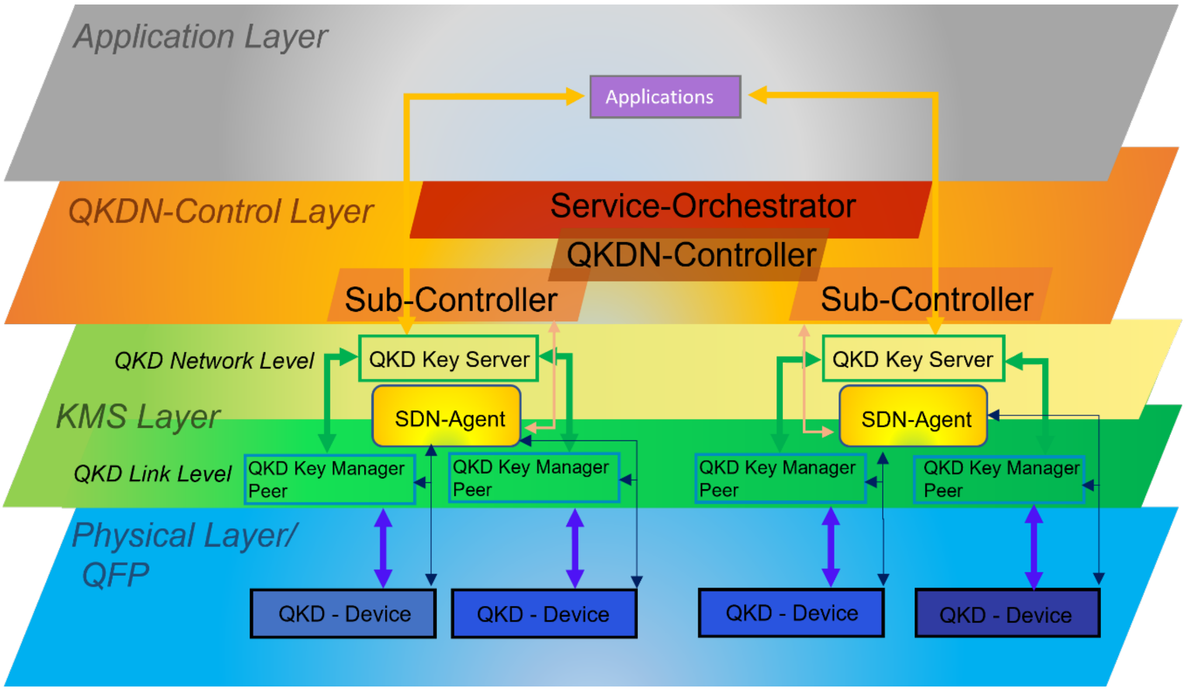

Figure 10: *Total QKDN according to ETSI*

Comparison with ITU-T

ITU-T also dedicates its recommendation suite Y.3800 ff to the topic of quantum key distribution and management in QKDN:

***ITU-T Y.3800*: Overview QKDN** [18]

This is where the foundations for a QKDN network are defined. Among other things, a QKDN must have the ability not only to distribute point-to-point keys, but also common multi-user/point keys to the requesting application in a specific format under special security requirements, and to control/manage them taking into account special QoS requirements and certain privacy policies. Here, too, the links between the key management systems, which are located locally at the same node, are often supported by QKDN-controllers and send corresponding parameters to them. The controller is familiar with activities such as key relay routing and control of QKD & key management links, as well as authentication control and QoS. In addition, there is a QKDN-manager who is responsible for FCAPS management (see Chapter ***Control systems in the (QKD) network***). There are strict key management policies for key lifetimes in the QKDN nodes (see Chapter ***The Key Management System (KMS) in the QKDN***); strict rules also apply in terms of independence from the applications, which means that the keys are only distinguished by their key-IDs and there is a mutual security demarcation between the application level/ the user network and the QKDN, so that the applications do not need any knowledge of the underlying QKDN processes. Conversely, the QKDN does not need any information about how the keys are used by the application and is limited to the knowledge of the key length and the associated application-ID.

***ITU-T Y.3803:* Key Management** [2]

The ITU-T recommendation for KMS differentiates between Key Management Agent (KMA) and Key Supply Agent (KSA). The KMA is tasked with QKD-related activities, such as receiving, storing, forwarding, managing and discarding according to policies during the life cycle of the keys; on the other hand, the KSA is responsible for the key requests, as well as the transmission of the KSA-key-(ID)s to the application layer and the metadata to the QKDN-manager (see ***Figure 11***, Interface *Ak*). Mutual authentication between KMA and KSA is required before a key request is passed on to the KMA. In addition, various *Key Relay* approaches are presented. If there is no direct KMS link between the KMSs, a route between the relays is requested by the QKDN-controller and the keys are routed accordingly to the destination node via relay nodes using XOR links at each node. (See Chapter ***The Key Management System (KMS) in the QKDN***)

***ITU-T Y.3804:* Control and management of QKDN** [9]

This guideline defines control and management in the QKDN context in more detail. While the basic functions are already described in Y.3800, routing, session control, QoS and FCAPS management functions are discussed in more detail here. In addition, reference points/interfaces between the control and management units are described, as well as their orchestration in the different layers and their cooperation with external management systems (e.g. User Network Management System). Each layer has its own control and management function, which corresponds to the QKDN management layer. The QKDN-controller controls the Physical Layer/ Quantum Layer and the Key Management Layer (but does not handle the keys themselves). In addition, it supports the QKDN Management Layer and the Application/Service Layer. In ***Figure 11*** the reference points associated with the controller are illustrated*: Ck, Cq, Cqrp (quantum relay point),* and *Cops (Optical Splitting),* while *Cx* is responsible for communication between the different controllers. *Mc* serves as an interface to the QKDN-manager and thus supports FCAPS. The QKDN-manager consists of quantum, key management, control layer management and a cross-layer management orchestration module spanning these three modules. The QKDN-manager manages the entire QKDN (FCAPS), has complete knowledge of the QKDN topology and regulates fault management, e.g. in the event of failure of QKD links. About the *Mu* interface, the QKDN Manager exchanges information with the User Network Management. (See Chapter ***Control systems in the (QKD) network***)

ITU-T Y.3805: **SDN-based control** [10]

This standard deals with the hierarchical structuring of SDN-controllers (see Chapter ***Control systems in the (QKD) network***). In addition, a general procedure for SDN control in QKDNs is introduced, consisting of different phases:

- "Service Request System Initialization Phase"
- "Key Generation Phase"
- "Key Request, Relay and Supply Phase"
- "Management Monitor Phase"
- "QKDN Virtualization Phase"

Compared to Y.3804, the control of programmable elements, application registration, topology acquisition of subordinated controllers and virtualization of the QKDN are added here; in addition, communication between the controllers must also be made possible. The interface *Ac* from ***Figure 11*** has been added here later to enable communication between the SDN controller and the application.

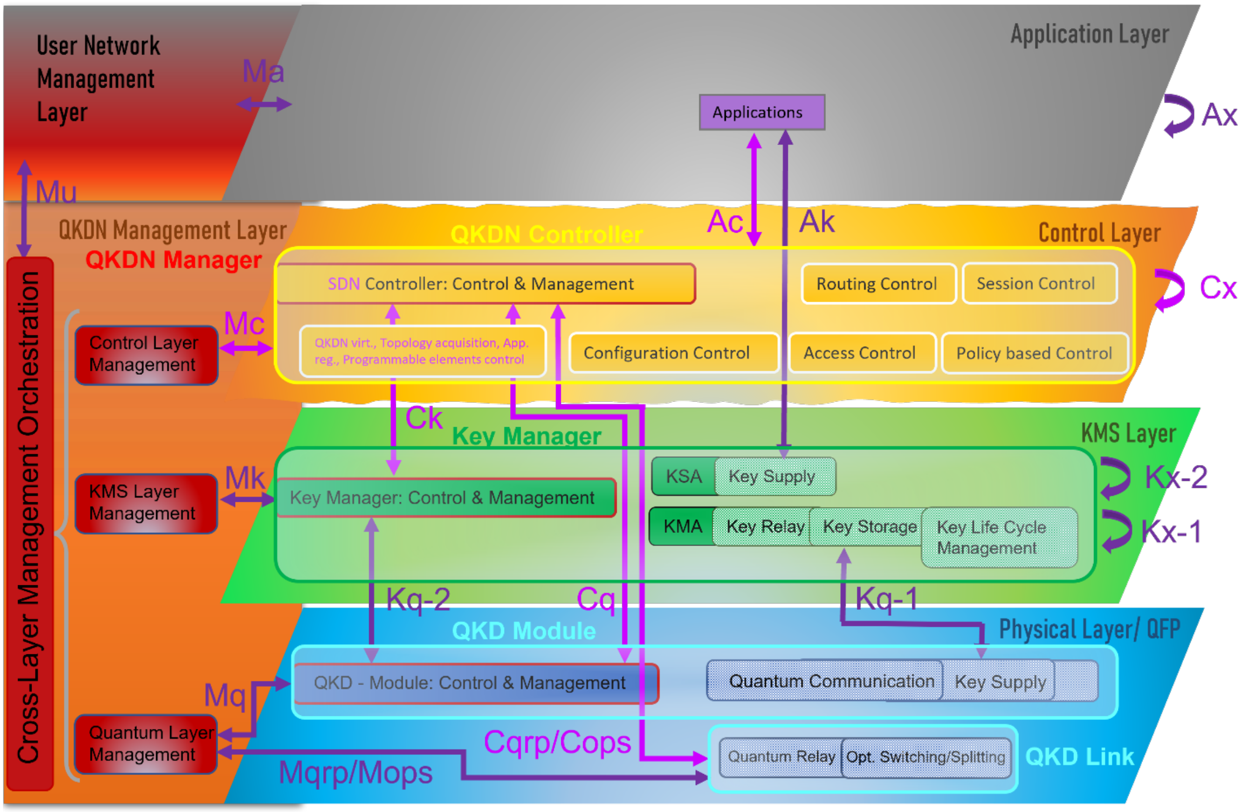

Figure 11: *The different layers of a QKDN with interfaces and functions in ITU-T Y.3804/5*

Significant differences in the description of a QKDN in ITU-T compared to ETSI are:

- The ITU defines cross-layer management orchestration with parallel management functions in the individual layers. ETSI, on the other hand, describes higher-level SDN-controllers/orchestrators with an integrated monitoring management function.
- The QKDN-manager at ITU also coordinates directly the QKD & KMS (& Control) layer individually, which are jointly controlled by the SDN-agent at ETSI.
- The direct interface between SDN-controller and application (*Ac*) currently only exists in ITU.

4. Key Monitoring Parameters

There are a variety of parameters in a QKDN that can or must be monitored. At the various interfaces, it should be possible to transmit various key monitoring parameters in order to be able to implement automatic monitoring for the security, stability and optimization of the network. In the KMS parameter area, this includes the fill levels of the key buffers, the fill rates, the average requested key rate (Secret Key Rate (SKR)) and the key recycling interval after the time to live (TTL) of the key has expired [13], [19]. Other parameters that can be determined in connection with the use of SDN in a testbed are the latency of the key distribution, the success of the key distribution and the average key consumption, for example, as shown in [20].

Currently, it is often difficult to obtain concrete information from the manufacturers on parameters that are determined by QKD devices and KMSs and how they are provided. The following parameters in connection with KMS and SDN would be of particular interest to be recorded in a QKDN:

- Average key stock needed in terms of buffer size considering fill rates:
    - Determination of threshold values at which an alarm should be raised if the level is too low -> new production of keys
    - Definition of a failure scenario, e.g., determining which data must definitely be encrypted and where switching to other methods for key generation (e.g. PQC) is possible
- Required SKR as needed in the network (application & time dependent)
- Key Rotation Interval (TTL of the key at application [13])

  In [21] different key rotation intervals of 1, 5, 15 and 60 min were tested with TTL = 4h and buffer size = 1000 keys. In the 1-minute interval, the success rate of the Quantum Key Rotation was 89%.
- Recording statistics of key requests (e.g., number and length of keys, link status) for optimization by the SDN:

  After ***ETSI GS QKD 015*** the nodes should provide information to the controller; e.g., for the registration and optimization of key management
- Additional useful parameters:
    - Latency, time synchronization and time constraints
    - Key-ID for identification not always transferable by default in the protocol (-> ***ETSI GS QKD 014*** or ***ITU-T Y.3803*** support) [22]
    - Introduction of further QoS parameters that can be actively and individually defined by the user (beyond ***ETSI GS QKD 004*** (e.g. only keys from/via certain KMS locations of certain manufacturers [1]).

Summary

This document describes the current situation with regard to QKD key management and the necessary control systems in QKDNs. For this purpose, the document has presented the function assignments of the KMS and SDN components and brought them together in a unifying manner in order to facilitate the targeted use of the securely generated QKD keys in the network in the future.

For this purpose, the existing standards of ETSI and ITU in connection with the two components were also presented. However, it is noticeable that the approaches of ETSI and ITU differ at this connection between key generation and secure key use and sometimes leave open questions regarding the realization of a functioning quantum communication network. There are still some important functions in the standards to be expanded or even missing in the field of network management systems but also of the KMS, such as:

- a KMS-to-KMS interface of different vendors between different nodes (without SDN)
- the possibility of direct communication between the SDN-controller and the application (in the case of ETSI)
- clear separation of the responsibilities of the control and management processes (combined in SDN in the future)
- in particular, it must be clarified whether the envisioned hybridization of the various keys (encryption of the QKD key with PQC & detection of emptying buffers by a Denial of Service attack) is part of the KMS [23].

In addition, it is often difficult to transfer additional (monitoring) parameters (e.g., key ID, time constraints) that are not provided in all standards and are not supported by all vendors.

It is therefore important to follow closely further developments in this very active area. Since the interaction of the various components in particular is difficult to investigate in theory, practical investigations in testbeds with many different manufacturers and components, such as MadQCI, will be very important in order to be able to use the keys in a QKD network efficiently and safely and thus enable the practical use of QKD systems.

Bibliography

[1] "Quantum-encrypted communication without borders", Oct. 2021, Accessed: June 28, 2024. Available at: https://www.stmd.bayern.de/wp-content/uploads/2022/03/Bayern-Oesterreich-Studie-Quantenverschluesselte-Kommunikation.pdf

[2] *Y.3803 : Quantum key distribution networks - Key management*. Accessed: June 27, 2024. Available at: https://www.itu.int/rec/T-REC-Y.3803-202012-I/en

[3] C. Lee, Y. Kim, K. Shim, and W. Lee, "Key-count differential-based proactive key relay algorithm for scalable quantum-secured networking", *J. Opt. Commun. Netw., JOCN*, vol. 15, no. 5, pp. 282–293, May 2023, doi: 10.1364/JOCN.478620.

[4] V. Martin *inter alia.*, "MadQCI: a heterogeneous and scalable SDN QKD network deployed in production facilities.", 2023, doi: https://doi.org/10.48550/arXiv.2311.12791.

[5] T. Choi, S. Yoon, T. Y. Kim, and H. Kim, "Design and Implementation of Quantum Key Distribution Network Control and Management", in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct. 2021, pp. 724–727. doi: 10.1109/ICTC52510.2021.9621170.

[6] P. James, S. Laschet, S. Ramacher, and L. Torresetti, "Key Management Systems for Large-Scale Quantum Key Distribution Networks", in *Proceedings of the 18th International Conference on Availability, Reliability and Security*, Benevento Italy: ACM, Aug. 2023, pp. 1–9. doi: 10.1145/3600160.3605050.

[7] M. Mehic, S. Rass, E. Dervisevic, and M. Voznak, "Tackling Denial of Service Attacks on Key Management in Software-Defined Quantum Key Distribution Networks", *IEEE Access*, vol. 10, pp. 110512–110520, 2022, doi: 10.1109/ACCESS.2022.3214511.

[8] E. Dervisevic *inter alia.*, "Simulations of Denial of Service Attacks in Quantum Key Distribution Networks", in *2022 XXVIII International Conference on Information, Communication and Automation Technologies (ICAT)*, June 2022, pp. 1–5. doi: 10.1109/ICAT54566.2022.9811238.

[9] *Y.3804 : Quantum key distribution networks - Control and management*. Accessed: July 22, 2024. Available at: https://www.itu.int/rec/T-REC-Y.3804/en

[10] *Y.3805 : Quantum key distribution networks - Software-defined networking control*. Accessed: July 22, 2024. Available at: https://www.itu.int/rec/T-REC-Y.3805/en

[11] O. Maurhart, T. Länger, A. Poppe, C. Pacher, M. Stierle, and H. Leopold, "Standardization and Certification of QKD-Devices and QKD Networks", Accessed June 27, 2024. Available at: https://2020.qcrypt.net/posters/QCrypt2020Poster133Maurhart.pdf

[12] M. Peev *inter alia.*, "The SECOQC quantum key distribution network in Vienna", *New J. Phys.*, Vol. 11, No. 7, p. 075001, July 2009, doi: 10.1088/1367-2630/11/7/075001.

[13] *ETSI GS QKD 004 Quantum Key Distribution (QKD); Application Interface*, August 2020. Available at: https://www.etsi.org/deliver/etsi_gs/QKD/001_099/004/02.01.01_60/gs_QKD004v020101p.pdf

[14] *ETSI GS QKD 014 Quantum Key Distribution (QKD); Protocol and data format of REST-based key delivery API*, February 2019. Available at: https://www.etsi.org/deliver/etsi_gs/QKD/001_099/014/01.01.01_60/gs_QKD014v010101p.pdf

[15] *ETSI GS QKD 015 Quantum Key Distribution (QKD); Control Interface for Software Defined Networks*, April 2022. Available at: https://www.etsi.org/deliver/etsi_gs/QKD/001_099/015/02.01.01_60/gs_QKD015v020101p.pdf

[16] *ETSI GS QKD 018 Quantum Key Distribution (QKD); Orchestration Interface for Software Defined Networks*, April 2022. Available at: https://www.etsi.org/deliver/etsi_gs/QKD/001_099/018/01.01.01_60/gs_QKD018v010101p.pdfhttps://www.etsi.org/deliver/etsi_gs/QKD/001_099/018/01.01.01_60/gs_QKD018v010101p.pdf

[17] *ETSI DGS QKD 020 InteropKMS*, December 1, 2024. Available at: https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=63115

[18] *Y.3800 : Overview on networks supporting quantum key distribution*. Accessed: July 22, 2024. Available at: https://www.itu.int/rec/T-REC-Y.3800/en

[19] G. Pmo and P. Hasleham, "QKD Concepts and Considerations", Accessed: 28 June 2024. Available at: https://resources.geant.org/wp-content/uploads/2024/02/GN5-1_White-Paper_QKD-Concepts-and-Considerations.pdf

[20] Y. Wang, X. Yu, Z. Wang, Y. Cao, Y. Zhao, and J. Zhang, "Demonstration of Hierarchical SDN Orchestration for End-to-End Key Provisioning in Large-Scale Quantum Key Distribution Networks", in *2023 21st International Conference on Optical Communications and Networks (ICOCN)*, Qufu, China: IEEE, July 2023, pp. 1–4. doi: 10.1109/ICOCN59242.2023.10236303.

[21] N. Makris *inter alia.*, "Field demonstration of a fully managed, L1 encrypted 3- node network with hybrid relayed-QKD and centralized symmetric classical key management", [Online]. Available at: https://arxiv.org/pdf/2403.08526

[22] "Validation of a Quantum Safe MACsec Implementation.pdf". Accessed: June 4, 2024. Available at: https://www.juniper.net/content/dam/www/assets/white-papers/us/en/2022/validation-of-quantum-safe-macsec-white-paper.pdf

[23] "EuroQCI ConOps (Concept of Operations) | Shaping Europe's digital future". Accessed: December 4, 2024. Available at: https://digital-strategy.ec.europa.eu/en/miscellaneous/euroqci-conops-concept-operations

List of Figures

List of Tables

Appendix

Compilation of references to QKDN standards from KMS & SDN

ETSI:

[A1]
https://www.etsi.org/deliver/etsi_gs/QKD/001_099/004/02.01.01_60/gs_QKD004v020101p.pdf

[A2]
https://www.etsi.org/deliver/etsi_gs/QKD/001_099/014/01.01.01_60/gs_QKD014v010101p.pdf

[A3]
https://www.etsi.org/deliver/etsi_gs/QKD/001_099/015/02.01.01_60/gs_QKD015v020101p.pdf

[A4]
https://www.etsi.org/deliver/etsi_gs/QKD/001_099/018/01.01.01_60/gs_QKD018v010101p.pdf

ITU-T:

[A5]
https://www.itu.int/rec/T-REC-Y.3800/en

[A6]
https://www.itu.int/rec/T-REC-Y.3803-202012-I/en

[A7]
https://www.itu.int/rec/T-REC-Y.3804/en

[A8]
https://www.itu.int/rec/T-REC-Y.3805/en

Secondary literature:

[B1] Key Management Systems for Large-Scale Quantum Key Distribution Networks
https://dl.acm.org/doi/10.1145/3600160.3605050

[B2] Interoperable KMS
https://www.mdpi.com/1099-4300/25/6/943

[B3] Standardization
https://2020.qcrypt.net/posters/QCrypt2020Poster133Maurhart.pdf

[B4] Standardization progress
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8596065