INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 2022-2024

SG13-TD38/WP3

STUDY GROUP 13

Original: English

Question(s): 16/13 Geneva, 4-15 July 2022

TD

Source: Editors

Title: Draft Recommendation ITU-T Y.QKDN-ml-fra: "Quantum key distribution networks - functional requirements and architecture for machine learning"

Contact:	Qingcheng Zhu Beijing University of Posts and Telecommunications. China	Tel: +86-18611519588 E-mail: qingcheng@bupt.edu.cn
Contact:	Yongli Zhao Beijing University of Posts and Telecommunications. China	Tel: +86-10-61198108 E-mail: yonglizhao@bupt.edu.cn
Contact:	Xiaosong Yu Beijing University of Posts and Telecommunications. China	Tel: +86-10-61198108 E-mail: xiaosongyu@bupt.edu.cn
Contact:	Zhangchao Ma CAS Quantum Network Co., Ltd. China	Tel: +86-10-83057625 E-mail: mazhangchao@casquantumnet.com
Contact:	Junsen Lai China Academy of Information and Communication Technology (CAICT), MIIT	Tel: +86-10-62300592 E-mail: laijunsen@caict.ac.cn
Contact:	Taesang Choi ETRI Republic of Korea	Tel: +82-10-2740-5628 E-mail: choits@etri.re.kr

Abstract: This document includes the output of Recommendation ITU-T Y.QKDN-ml-fra "Quantum key distribution networks - functional requirements and architecture for machine learning".

Summary

This TD is the output document for the draft Recommendation ITU-T Y.QKDN-ml-fra "Quantum key distribution networks - functional requirements and architecture for machine learning" based on the following input contributions and the discussion during the Q16/13 meeting, 4 – 15 July 2022. As the discussion results, contents should be further improved.

C-0151R1	BUPT, CAS Quantum Network Co. Ltd., MIIT	Proposed improvements to ITU-T Y.QKDN-ml-fra (for consent)	Q16/13

- Proposal of contribution

− This contribution includes the revised contents based on discussion results for the draft Recommendation ITU-T Y.QKDN-ml-fra “Quantum key distribution networks - functional requirements and architecture for machine learning” based on the results of SG13 Q16 meeting (7 – 9 June 2022). It proposes the improvements for consent.

- Meeting result

− According to the discussion results, the document is not very stable now which is not suitable for consent and still needs further improvement. The definition of QKDNml, the functional requirements and the architecture need to be improved in the future.

Attachments:

Annex I: Draft Recommendation ITU-T Y.QKDN-ml-fra “Quantum key distribution networks - functional requirements and architecture to enable machine learning” (output of Q16/13, 4 – 15 July 2022)

Annex I
Draft Recommendation ITU-T Y.QKDN-ml-fra

Quantum key distribution networks - functional requirements and architecture to enable machine learning

Summary

QKDN is expected to be able to maintain stable operations and meet the requirements of various cryptographic applications ~~requirements in an~~ efficient~~-~~ly~~way~~. Due to the advantages of machine learning (ML) related to autonomous learning, ML can help to overcome the challenges of QKDN in terms of quantum layer performances, key management layer performances and QKDN control and management efficiency. Based on the functional requirements and architecture of QKDN in [ITU-T Y.3801] and [ITU-T Y.3802], this recommendation is to specify a framework for ML-enabled QKDN (QKDNml), including the role of ML in QKDN, the functional requirements, architecture and operational procedures ~~to enable ML in QKDN~~of QKDNml.

Keywords
Functional architecture; functional requirements; machine learning (ML); procedure; quantum key distribution (QKD); QKD network (QKDN).

Table of Contents

Draft Recommendation ITU-T Y.QKDN-ml-fra

Quantum key distribution networks - functional requirements and architecture to enable machine learning

1. Scope

This Recommendation specifies ~~the role of ML in QKDN, the functional requirements, architecture and procedures~~ a framework ~~to enable ML in QKDN~~for ML-enabled QKDN (QKDNml).

In particular, the Recommendation includes:

- Role of ML in QKDN;
- Functional requirements ~~for~~ of ~~ML-enabled QKDN~~QKDNml;
- Functional architecture ~~for~~ of ~~ML-enabled QKDN~~QKDNml;
- Operational ~~P~~procedures ~~to enable ML in QKDN~~of QKDNml.

This draft Recommendation specifies requirements for generic data collection. It does not specify the requirements for specific data related to PII.

2. References

[ITU-T Y.3800] Recommendation ITU-T Y.3800 (2019), *Framework for Networks to support Quantum Key Distribution*.

[ITU-T Y.3801] Recommendation ITU-T Y.3801 (2020), *Functional requirements for quantum key distribution networks*.

[ITU-T Y.3802] Recommendation ITU-T Y.3802 (2020), *Functional architecture of the Quantum Key Distribution network*.

~~[ITU-T Y.3803] Recommendation ITU-T Y.3803 (2020), *Key management for quantum key distribution network*.~~

~~[ITU-T Y.3804] Recommendation ITU-T Y.3804 (2020), *Control and Management for Quantum Key Distribution Network*.~~

~~[ITU-T Y. 3805] Recommendation ITU-T Y. 3805 (2021), *Software Defined Networking Control for Quantum Key Distribution Networks*~~

[ITU-T Y.3172] Recommendation ITU-T Y.3172 (2019), *Architectural framework for machine learning in future networks including IMT-2020*.

~~3. [ITU-T Y.3174] Recommendation ITU-T Y.3174 (2020), *Framework for data handling to enable machine learning in future networks including IMT-2020*.~~

~~4.~~3. Terms and definitions

~~4.1.~~3.1. Terms defined elsewhere

This recommendation uses the following terms defined elsewhere:

3.1.1 **key manager (KM)** [ITU-T Y.3800]: A functional module located in a quantum key distribution (QKD) node to perform key management in the key management layer.

3.1.2 **machine learning (ML)** [ITU-T Y.3172]: processes that enable computational systems to understand data and gain knowledge from it without necessarily being explicitly programmed.

NOTE - Definition adapted from [b-ETSI GR ENI 004].

3.1.3 **machine learning function orchestrator (MLFO)** [ITU-T Y.3172]: a logical orchestrator that can monitor and manage the nodes in a machine learning pipeline.

3.1.4 **machine learning model** [ITU-T Y.3172]: model created by applying machine learning techniques with data to learn from.

NOTE 1 – A machine learning model is used to generate predictions on new (untrained) data.

NOTE 2 – A machine learning model may be encapsulated in a deployable fashion in the form of a software or hardware component.

NOTE 3 – Machine learning techniques include learning algorithms (e.g. learning the function that maps input data attributes to output data).

3.1.5 **machine learning pipeline** [ITU-T Y.3172]: a set of logical nodes, each with specific functionalities, that can be combined to form a machine learning application in a telecommunication network.

NOTE – The nodes are entities that are managed in a standard manner and can be hosted in a variety of network functions.

3.1.6 **machine learning sandbox** [ITU-T Y.3172]: an environment in which machine learning models can be trained, verified and their effects on the network analysed.

NOTE – A machine learning sandbox is designed to prevent a machine learning application from affecting the network, or to restrict the usage of certain machine learning functionalities.

3.1.7 **network performance** [b-ITU-T E.417]: The performance of a portion of a telecommunications network that is measured between a pair of network-user or network-network interfaces using objectively defined and observed performance parameters.

3.1.8 **quantum key distribution (QKD)** [b-ETSI GR QKD 007]: Procedure or method for generating and distributing symmetrical cryptographic keys with information theoretical security based on quantum information theory.

3.1.9 **quantum key distribution link (QKD link)** [ITU-T Y.3800]: A communication link between two quantum key distribution (QKD) modules to operate the QKD.

NOTE – A QKD link consists of a quantum channel for the transmission of quantum signals, and a classical channel used to exchange information for synchronization and key distillation.

3.1.10 **quantum key distribution module (QKD module)** [ITU-T Y.3800]: A set of hardware and software components that implements cryptographic functions and quantum optical processes, including quantum key distribution (QKD) protocols, synchronization, distillation for key generation, and is contained within a defined cryptographic boundary.

NOTE – A QKD module is connected to a QKD link, acting as an endpoint module in which a key is generated. These are two types of QKD modules, namely, the transmitters (QKD-Tx) and the receivers (QKD-Rx).

3.1.11 **quantum key distribution network (QKDN)** [ITU-T Y.3800]: A network comprised of two or more quantum key distribution (QKD) nodes connected through QKD links.

NOTE – A QKDN allows sharing keys between the QKD nodes by key relay when they are not directly connected by a QKD link.

3.1.12 **quantum key distribution network controller (QKDN controller)** [ITU-T Y.3800]: A functional module, which is located in a quantum key distribution (QKD) network control layer to control a QKD network.

3.1.13 **quantum key distribution network manager (QKDN manager)** [ITU-T Y.3800]: A functional module, which is located in a quantum key distribution (QKD) network management layer to monitor and manage a QKD network.

3.1.14 **quantum key distribution node (QKD node)** [ITU-T Y.3800]: A node that contains one or more quantum key distribution (QKD) modules protected against intrusion and attacks by unauthorized parties.

NOTE – A QKD node can contain a key manager (KM). ~~TBD~~

3.1.15 **quality of service** [b-ITU-T Q.1743]: The collective effect of service performances, which determine the degree of satisfaction of a user of a service. It is characterized by the combined aspects of performance factors applicable to all services, such as: service operability performance; service accessibility performance; service retainability performance; service integrity performance; and other factors specific to service.

~~4.2.~~3.2. Terms defined in this Recommendation

This chapter defines all the terms used in this recommendation.

3.2.1 **Machine-learning-enabled quantum key distribution network (ML-enabled QKDN):** A quantum key distribution network (QKDN) that extends its functionalities enabled by machine learning (ML) capabilities of different objectives.

NOTE – ML is an optional functionality for QKDN ~~specific to optimization purposes.~~ ~~TBD~~

~~5.~~4. Abbreviations and acronyms

This chapters describes all the abbreviations and acronyms used in the recommendation.

C	Collector (ML pipeline)
D	Distribution (ML pipeline)
M	Model (ML pipeline)
ML	Machine learning
MLFO	Machine Learning Function Orchestrator
MLMS	Machine Learning Management Subsystem
P	Policy (ML pipeline)
PP	Pre-processor (ML pipeline)
QKD	Quantum key distribution
QKDN	Quantum key distribution network
QKDNml	ML-enabled QKDN
QKDN-opt	QKDN-optimization
RUL	Remaining Use Life
SRC	S~~s~~ource
SINK	Sink node

~~6.~~5. Conventions

In this Recommendation:

The keywords "is required to" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus this requirement need not be present to claim conformance.

The keywords "is not recommended" indicate a requirement which is not recommended but which is not specifically prohibited. Thus, conformance with this specification can still be claimed even if this requirement is present.

The keywords "can optionally" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

In the body of this Recommendation and its annexes, the words shall, shall not, should, and may sometimes appear, in which case they are to be interpreted, respectively, as is required to, is prohibited from, is recommended, and can optionally. The appearance of such phrases or keywords in an appendix or in material explicitly marked as informative is to be interpreted as having no normative intent.

~~7.~~6. Overview

~~Quantum key distribution network (~~QKDN~~)~~ is a technology that extends the reachability and availability of ~~quantum key distribution (~~QKD~~)~~, which is stated in [ITU-T Y.3800]. It is comprised of two or more QKD nodes connected through QKD links. In a QKDN, two or more designated parties in a user network can share the keys for various cryptographic applications. When the QKDN becomes large-scale~~ where there are multiple QKD nodes and links~~, the challenges will be faced in terms of quantum layer performances, key management layer performances and QKDN control and management efficiency~~ will be faced~~. QKDN is expected to be able to maintain stable operations and meet the requirements of various cryptographic applications ~~requirements~~ in an efficient way.

However, to improve QKDN performances, QKDN faces the following important challenges:

1) Without the awareness of sudden QKDN performance deterioration in advance, high cost (e.g. time cost, labour cost) and instability of QKDN will increase.

2) For the large amount of heterogenous data in QKDN, there is difficulty to accurately perceive the needed and valuable information for use, which will affect QKDN performances.

3) Since the requirements of cryptographic applications are various (e.g. different security requirements) and a large number of cryptographic applications arrive and leave dynamically, it is difficult to schedule the QKDN resources for cryptographic applications under the finite resource limit.

To overcome the above challenges, applying ML technology into QKDN is a promising solution. ML can extract implicit relationships between input and output data, and use this learnt mapping to analyse new data. It has been applied to networking field and can intelligently learn various network environments and react to dynamic situations ([ITU-T Y.3170]). In recent years, ML technologies based on neural networks have seen many developments in both hardware and software, and they have attracted attention from both the academia and the industry. There is also an increasing number of new low-power devices implementing on-board acceleration chips for neural networks.

Application cases of enabling ML in QKDN have been specified in [b-ITU-T Y-Suppl.70]. In the quantum layer of QKDN, ML can be applied to realize quantum channel performance prediction, QKD system parameter optimization and ~~remaining use life (~~RUL~~)~~ prediction of components in a QKD system; in the key management layer of QKDN, ML can be applied to realize intelligent key formatting, key storage management, and suspicious behavior detection; in the control and management layers of QKDN, ML can be applied in routing and QKDN fault prediction to improve control and management efficiency. There will be many benefits of enabling ML in QKDN.

Hence, due to the advantages of machine learning (ML) related to autonomous learning, ML can help to overcome the challenges of QKDN in terms of quantum layer performances, key management layer performances and QKDN control and management efficiency. A ~~ML-enabled QKDN~~QKDNml~~s~~ ~~are~~ is optimal to be constructed which extends QKDN functionalities by enabling ML capabilities of different objectives~~where ML is optional for improving QKDN performances~~. ML is an optional functionality for QKDN specific to different QKDN optimization purposes. Based on the functional requirements and architecture of QKDN in [ITU-T Y.3801] and [ITU-T Y.3802], this recommendation is to specify a framework for QKDNml, including the role of ML in QKDN, the functional requirements, architecture and operational procedures ~~for~~ of ~~ML-enabled QKDN~~QKDNml.

~~8.~~7. Functional requirements of~~for~~ ~~ML-enabled QKDN~~QKDNml

The functional requirements of QKDN have been specified in [ITU-T Y.3801]. In case of considering ML in QKDN, the additional functional requirements related to ML in the subclause are applied to realize the ~~ML-enabled QKDN~~QKDNml.

[Editor's note: The functional requirements of QKDNml need further improvement.]

7.1 High-level requirements ~~for~~ of ~~ML-enabled QKDN~~QKDNml

The high-level requirements ~~for~~ of ~~ML-enabled QKDN~~QKDNml are as follows.

- ~~QKDN~~QKDNml is required to support the data communication between ML-related functional components and other QKDN functional components defined in [ITU-T Y.3800];
- ~~QKDN~~QKDNml is required to support the configuration, management and orchestration for the ML-related functional components;
- ~~QKDN~~QKDNml is required to support the data collection, data pre-processing, data repository, modelling and training functions;
- QKDNml is required to support the ML model in QKDN for different QKDN optimization purposes~~ML model applying~~;
- ~~QKDN~~QKDNml is recommended to use the existing reference points in QKDN defined in [ITU-T Y.3802] and extend them with reference points specific to ML requirements.~~or extending the reference points to realize the data communication;~~

7.2 Functional requirements ~~for~~ of ~~ML-enabled QKDN~~QKDNml data collection

The QKDN data can be collected from the quantum layer, key management layer, QKDN control layer, QKDN management layer, service layer and user network management layer either passively or actively.

~~The functional requirements for quantum layer data collection are as follows.~~

- ~~QKDN~~ Data collection function in QKDNml is required to collect static and dynamic QKDN data from quantum layer, key management layer, QKDN control layer and QKDN management layer.

NOTE 1 - Static QKDN data is collected from quantum layer such as parameters of QKD modules, history status information of QKD modules; dynamic QKDN data is collected from quantum layer such as quantum bit error rates, key generation rate, performance information of QKD modules.

NOTE 2 - Static QKDN data is collected from key management layer such as history quantum layer data set; dynamic QKDN data is collected from key management layer such as status of key storage and status of key authentication.

NOTE 3 - Static QKDN data is collected from QKDN control layer such as parameters of QKD modules and history status information of QKD modules; dynamic QKDN data is collected from QKDN control layer such as routing and rerouting information and status of resource allocation.

NOTE 4 - Static QKDN data is collected from QKDN management layer such as history data of fault management, history data of configuration and history data of security management; dynamic QKDN data is collected from QKDN management layer such as multi-layer resource usage data and multi-layer performance data.

- Data collection function in QKDNml can optionally collect static and dynamic QKDN data from service layer and user network management layer.

NOTE 1 - Static QKDN data is collected from service layer such as history cryptographic application information; dynamic QKDN data is collected from service layer such as current cryptographic applications information.

~~•~~NOTE 2 - Static QKDN data is collected from user network management layer such as history user requirements; dynamic QKDN data is collected from user network management layer such as current user requirements. ~~(e.g., parameters of QKD modules, history status information of QKD modules).~~

~~QKDN is required to collect dynamic QKDN data from quantum layer (e.g., quantum bit error rates, key generation rate, performance information of QKD modules).~~

~~The functional requirements for key management layer data collection are as follows.~~

~~QKDN is required to collect static QKDN data from key management layer (e.g., history quantum layer data set).~~

~~QKDN is required to collect dynamic QKDN data from key management layer (e.g., status of key storage, status of key authentication).~~

~~The functional requirements for QKDN control layer data collection are as follows.~~

~~QKDN is required to collect static QKDN data from QKDN control layer (e.g., network topology).~~

~~QKDN is required to collect dynamic QKDN data from QKDN control layer (e.g., routing and rerouting information, status of resource allocation).~~

~~The functional requirements for QKDN management layer data collection are as follows.~~

~~QKDN is required to collect static QKDN data from QKDN management layer (e.g., history data of fault management, history data of configuration, history data of security management).~~

~~QKDN is required to collect dynamic QKDN data from QKDN management layer (e.g., multi-layer resource usage data, multi-layer performance data).~~

~~The functional requirements for service layer data collection are as follows.~~

~~• QKDN can optionally collect static QKDN data from service layer (e.g., history cryptographic application information).~~

~~• QKDN can optionally collect dynamic QKDN data from service layer (e.g., current cryptographic applications information).~~

~~NOTE – Cryptographic application information can optionally include the arriving time, the duration time, the required security levels.~~

~~The functional requirements for user network management layer data collection are as follows.~~

~~• QKDN can optionally collect static QKDN data from user network management layer (e.g., history user requirements).~~

~~• QKDN can optionally collect dynamic QKDN data from user network management layer (e.g., current user requirements).~~

7.3 Functional requirements for ~~ML-enabled QKDN~~QKDNml data pre-processing

- Data pre-processing function in QKDNml ~~QKDN~~ is required to perform extract-transform-load ~~(ETL)~~ and transform the collected multi-source, heterogeneous QKDN raw data into understandable, unified and easy-to-use structures.
- Data pre-processing function in QKDNml ~~QKDN~~ is required to clean and filter noisy data from the collected heterogeneous QKDN raw data.
- Data pre-processing function in QKDNml ~~QKDN~~ is recommended to normalize and unify the data format of the collected heterogeneous QKDN raw data for further storage and analysis.

7.4 Functional requirements for history QKDN data repository

~~•~~ Data repository function in QKDNml ~~QKDN~~ is required to store ~~a large amount of~~ heterogeneous QKDN pre-processed data.

- ~~QKDN control and management layers are recommended to support history QKDN data repository.~~

7.5 Functional requirements for modelling and training

- Modelling and training function in QKDNml ~~QKDN~~ is required to construct ML models based on the pre-processed QKDN data.
- Modelling and training function in QKDNml ~~QKDN~~ is recommended to train the ~~machine learning~~ML models based on the available pre-processed QKDN data.
- QKDNml is recommended to support the realization of modelling and training function independent on QKDN.

7.6 Functional requirements for ML model applying

- ~~QKDN is required to select ML models.~~
- ~~QKDN~~QKDNml is recommended to support applications of ML models in quantum layer, key management layer, QKDN control layer and QKDN management layer.

7.7 Functional requirements for ML configuration and orchestration

- ~~The QKDN controller~~QKDNml is ~~required~~ required to support configuration and orchestration of ~~all~~ ML-related functional components by using or extending the existing QKDN reference points or defining specific APIs.

~~•~~

~~9. The QKDN manager is required to orchestrate all ML-related functional components.~~

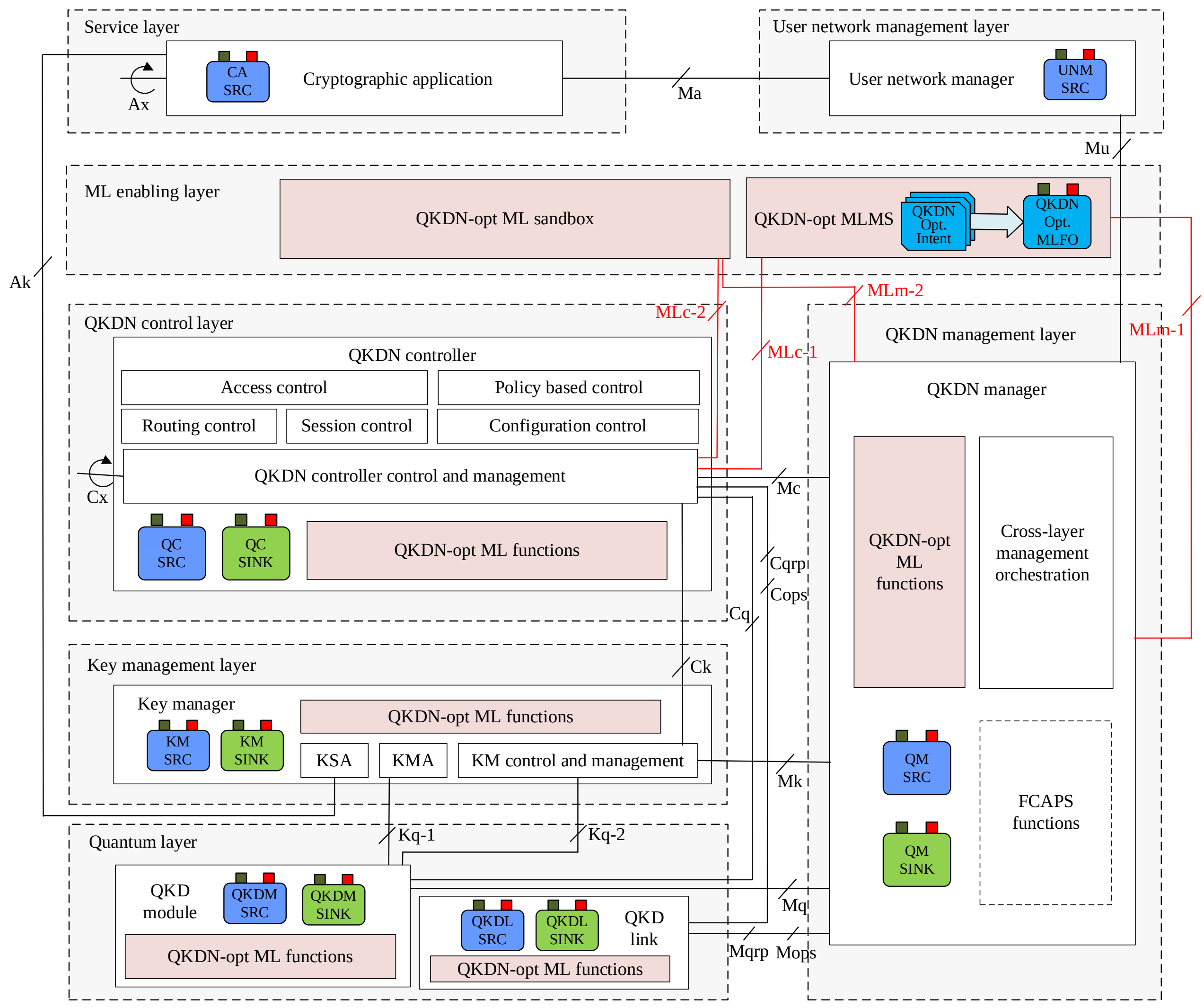
~~10.~~8. Functional architecture ~~for~~ of ~~ML-enabled QKDN~~QKDNml

ML is enabled in QKDN to optimize the performances of QKDN such as quantum channel performance, QKD system parameter, key management efficiency and fault diagnosis. Based on the functional architecture of QKDN in [ITU-T Y.3800] and [ITU-T Y.3802] and the ML pipeline defined in [ITU-T Y.3172], the functional architecture model ~~for~~of ~~ML-enabled QKDN~~QKDNml is designed as shown in Fig. 8.1. The architecture model includes the following ML-related architectural essence in the ~~ML-enabled QKDN~~QKDNml:

- High-level architectural ML-related functional components defined in [ITU-T Y.3172]: ML pipeline, MLFO and ML sandbox.
- ML-related functional components in ~~ML-enabled QKDN~~QKDNml~~s~~ defined in this Recommendation: ~~QKDN optimization (~~QKDN-opt~~)~~ ML pipeline, QKDN-opt ML sandbox, and QKDN-opt ~~machine learning management subsystem (~~MLMS~~)~~.

A new ML enabling layer including ~~T~~the QKDN-opt ML ~~functions~~MS and ML sandbox ~~are located in QKDN control layer and the QKDN-opt MLMS is located in QKDN management layer~~is added based on the QKDN architecture in [ITU-T Y.3802]. The QKDN-opt ML pipeline subsystems are integrated with QKDN functional components in different QKDN layers, which are able to realize different ML applications for QKDN optimization. The functional components in the ML enabling layer may need a large amount of resources such as CPU and GPU for working. The realization of ML enabling layer is flexible based on the constraints defined in the declarative specifications of the ML applications ([ITU-T Y.3172]). The ML enabling layer is able to communicate with QKDN control layer and QKDN management layer using the reference points MLc and MLm, so as to transfer data for training or testing models among ML functionalities or orchestrating QKDN-opt ML pipeline subsystems. ~~The data communication between these ML-related functional components is implemented by the existing reference points defined in [ITU-T Y.3802].~~

[Editor’s note: The location of the ML enabling layer and the related reference points needs the further consideration.]



🟩 Service for ingress (producer) SRC: source of data; SINK: target of ML output; MLFO: ML function orchestrator; MLMS: ML management system;
🟥 Service for egress (consumer) FCAPS: fault, configuration, accounting, performance and security;

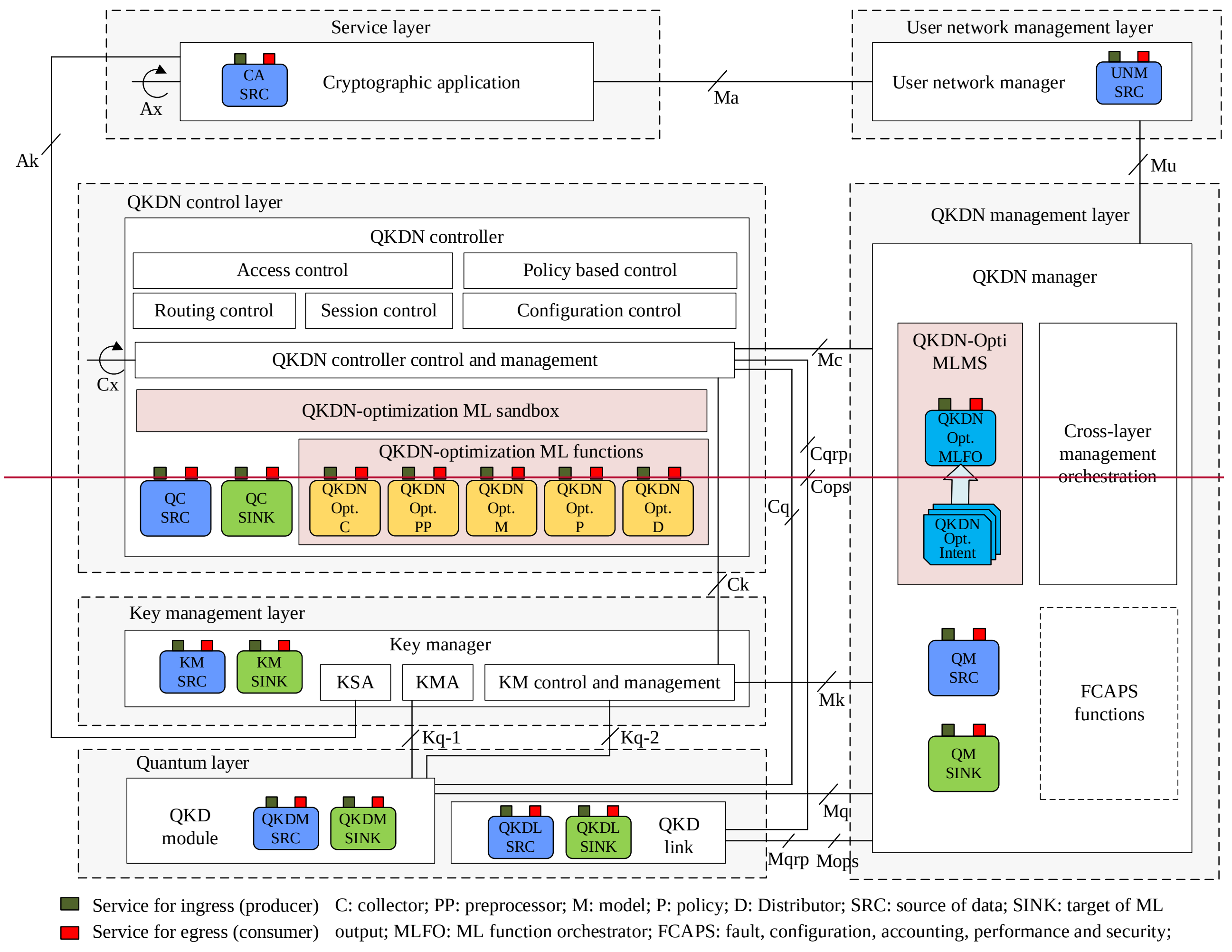


Fig. 8.1. Functional architecture model ~~for~~of ~~ML-enabled QKDN~~QKDNml

8.1 QKDN-opt~~imization~~ ML pipeline

The ~~QKDN-optimization (~~QKDN-opt~~)~~ ML pipeline is a set of logical nodes, each with specific functionalities that can be combined to form an ML application in ~~ML-enabled QKDN~~QKDNml. The symbol used to denote the service ingress and egress points of the ML pipeline nodes is illustrated in [ITU-T Y.3172]. The QKDN-opt ML pipeline has three parts including the QKDN-opt source, QKDN-opt ML functions and QKDN-opt target. The ML functions are able to collect input data from ~~source of data (SRC) nodes~~sources (SRCs) ([ITU-T Y.3172]) in different layers of QKDNs. The SINK is the target of ~~The target of~~ the ML output ([ITU-T Y.3172]) ~~(SINK)~~ can be the elements in quantum layer, key management layer and QKDN control and management layers. More details related to ML pipeline subsystems can be found in [ITU-T Y.3172].

8.1.1 QKDN-opt ~~imization source (~~SRC~~)~~

QKDN-opt ~~source~~ SRC is the source of QKDN data that can be used as input to the ML applications in ~~ML-enabled QKDN~~QKDNml. The types of SRCs include ([ITU-T Y.3800]):

– QKD-module source (QKDM-SRC) is responsible for reporting the data (static, dynamic) from QKD modules in the quantum layer to the QKDN-opt collector;

– QKD-link source (QKDL-SRC) is responsible for reporting the data (static, dynamic) from QKD links in the quantum layer to the QKDN-opt collector;

– Key-manager (KM-SRC) source is responsible for reporting the data (static, dynamic) from the key manager in the key management layer to the QKDN-opt collector;

- QKDN-controller source (QC-SRC) is responsible for reporting the data (static, dynamic) from the QKDN controller in the QKDN control layer to the QKDN-opt collector;
- QKDN-manager source (QM-SRC) is responsible for reporting the data (static, dynamic) from the QKDN manager in the QKDN management layer to the QKDN-opt collector;
- Cryptographic-application source (CA-SRC) is responsible for reporting the cryptographic application data (static, dynamic) in the service layer to the QKDN-opt collector;
- User-network-manager source (UNM-SRC) is responsible for reporting the user requirement data (static, dynamic) in the user network management layer to the QKDN-opt collector;

8.1.2 QKDN-opt~~imization~~ ML functions

The QKDN-opt ML functions support a set of functional elements in a ML pipeline subsystem including collector (C), pre-processor (PP), model (M), policy (P) and distributor (D).

- **QKDN-opt~~imization collector (~~C~~)~~**

QKDN-opt C is responsible for collecting QKDN data from one or more QKDN-opt SRCs. A C may have the capability to configure SRC nodes. Such configurations may be used to control the nature of data, its granularity and periodicity while it is generated from the QKDN-opt SRCs.

- **QKDN-opt~~imization pre-processing (~~PP~~)~~**

QKDN-opt PP is responsible for cleaning, aggregating, normalizing or performing any other PP of heterogeneous data that should be in a suitable form so that the QKDN-opt~~imization~~ ML model can consume it.

- **QKDN-opt~~imization~~ ML ~~model (~~M~~)~~**

QKDN-opt ML M is responsible for deploying the trained ML models for different ~~ML applications in~~ QKDN optimization purposes.

- **QKDN-opt~~imization policy (~~P~~)~~**

QKDN-opt P is responsible for making QKDN P decisions based on the results of the QKDN-opt ML M.

- **QKDN-opt~~imization distribution (~~D~~)~~**

QKDN-opt D is responsible for identifying the SINK(s) and distributing the QKDN-opt P decisions to the corresponding SINK nodes.

8.1.3 QKDN-opt~~imization target (~~SINK~~)~~

QKDN-opt SINK is the target of the ML output ~~of ML applications~~ in ~~ML-enabled QKDN~~QKDNml on which actions are taken. The types of ~~SRCs~~ SINKs can include:

- QKD-module SINK (QKDM- SINK) represents that the QKD module is the target of ~~the output of~~ configurations (as a result of ML pipeline execution)~~ML applications~~ in the quantum layer;

  NOTE - ML output is applied to QKD modules to optimize quantum layer performances of QKDN. The use cases can include ML-based QKD system parameter optimization and ML-based ~~remaining use life (~~RUL~~)~~ prediction of components in a QKD system ([b-ITU-T Y-Suppl.70]).

- QKD-link SINK (QKDL- SINK) represents that the QKD link is the target of ~~the output of ML applications~~configurations in the quantum layer;

  NOTE - ML output is applied to QKD links in the quantum layer of a QKDN to optimize quantum layer performances of QKDN. A use case is ML-based quantum channel performance prediction ([b-ITU-T Y-Suppl.70]).

- Key-manager (KM- SINK) SINK represents that key manager is the target of configurations ~~the target of the output of ML applications~~ in the key management layer;

  NOTE - ML output is applied to the key manager in the key management layer to optimize key management efficiency and stability. Three use cases are ML-based key formatting, ML-based key storage management, and ML-based suspicious behavior detection in the key management layer ([b-ITU-T Y-Suppl.70]).

- QKDN-controller SINK (QC- SINK) represents that QKDN controller is the target of configurations ~~the target of the output of ML applications~~ in the QKDN control layer;

  NOTE – ML output is applied to the QKDN controller in the QKDN control layer to improve QKDN control efficiency. Two use cases are ML-based data collection and data pre-processing and ML-based routing ([b-ITU-T Y-Suppl.70]).

- QKDN-manager SINK (QM- SINK) represents that QKDN manager is the target of configurations ~~the target of the output of ML applications~~ in the QKDN management layer to the QKDN-related collector;

  NOTE - ML output is applied to QKDN manager in the QKDN management layer to optimize QKDN management efficiency. A use case is ML-based QKDN fault prediction ([b-ITU-T Y-Suppl.70]).

8.2 QKDN-opt~~imization~~ ML sandbox

A QKDN-opt ML sandbox is an isolated domain that allows the hosting of separate ML pipelines to train, test and evaluate them before deploying them in a QKDN. The QKDN-opt ML sandbox subsystem allows network operators to study the effect of ML outputs before deploying them on live QKDNs. For training or testing, the QKDN-opt ML sandbox can use data generated from a simulated ML underlay QKDN.

8.3 QKDN-opt~~imization~~ ML~~ management subsystem~~MS

This QKDN-opt MLMS includes QKDN-opt intent, MLFO and other management and orchestration functions.

- **QKDN-opt~~imization~~ intent**

QKDN-opt intent is a declarative description ([ITU-T Y.3172])~~that is used to specify an ML application~~ in QKDN. QKDN-opt intent provides a basis for mapping ML use cases to different layers in QKDN.

NOTE – Intent is a declarative description which is used to specify a ~~machine learning~~ML application. Intent does not specify any technology-specific network functions to be used in the ML application and provides a basis for mapping ML use cases to diverse technology-specific instantiations. Intent can use a meta language specific for machine learning to define ML applications. ([ITU-T Y.3172])

- **QKDN-opt~~imization~~ MLFO**

The QKDN-opt MLFO is a logical node with functionalities that manage and orchestrate the nodes of the QKDN-opt ML pipelines and QKDN-opt ML sandbox based on QKDN-opt intent or dynamic QKDN conditions. The MLFO provides chaining functionality, i.e., connecting ML nodes together to form an ML pipeline. The MLFO selects the ML model based on the QKDN-opt intents, corresponding QKDN capabilities and constraints of the ~~ML application~~QKDN optimization.

NOTE – For example, chaining can be used to connect an SRC ~~node~~ instantiated in the quantum layer with collector and PP nodes instantiated in the QKDN control layer. The chain itself is declared in the use case specification and its technology-specific implementation in the network is done by the MLFO. The MLFO determines the chaining considering the constraints (e.g., timing constraints for prediction). ([ITU-T Y.3172])

8.4 Reference points

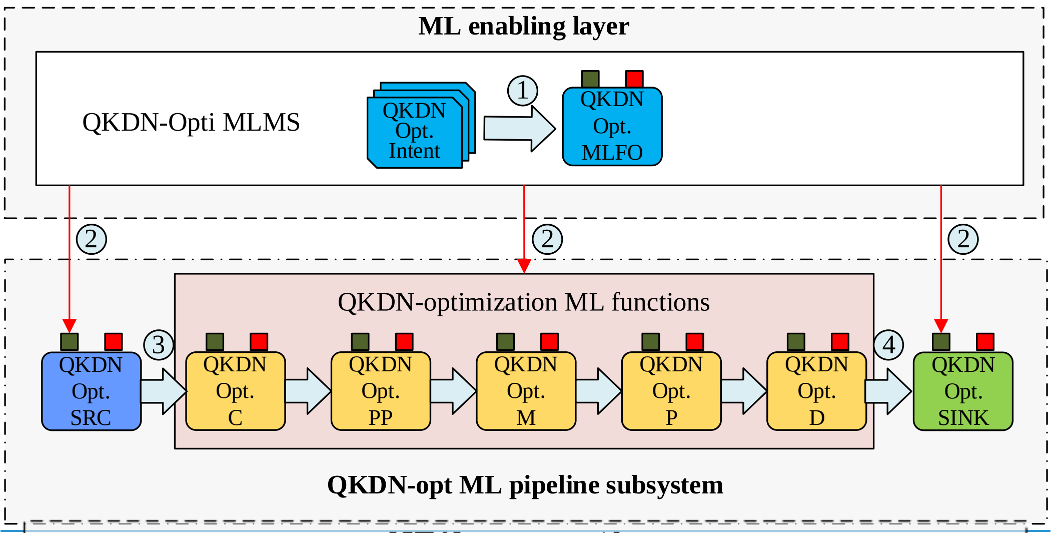
The following reference points are relevant to connections between ML enabling layer and QKDN layers, so as to realize the ML application for QKDN optimization.

[Editor's note: the description of the ML-related reference points needs the further improvement.]

- **MLc-1:** a reference point connecting QKDN-opt MLMS and QKDN controller control and management. It is responsible for data communication between QKDN-opt MLMS and QKDN controller.
- **MLc-2:** a reference point connecting QKDN-opt ML sandbox and QKDN controller control and management. It is responsible for data communication between QKDN-opt ML sandbox and QKDN controller.
- **MLm-1:** a reference point connecting QKDN-opt MLMS and QKDN manager. It is responsible for data communication between QKDN-opt MLMS and QKDN manager.
- **MLm-2:** a reference point connecting QKDN-opt ML sandbox and QKDN manager. It is responsible for data communication between QKDN-opt ML sandbox and QKDN manager.

~~11.~~9. Operational ~~P~~procedures ~~to enable ML in QKDN~~ of QKDNml~~.~~

A QKDN-opt ML pipeline is a set of logical nodes that can be combined ~~to form an ML application~~ for QKDN performance optimization. Based on ML correlation between QKDN data and the QKDN performances to be optimized in different layers of QKDN, QKDN-opt M can classify or predict the QKDN performance values according to the specific QKDN-opt intents. Then, the actions for QKDN-opt SINK are taken to realize the QKDN performance optimization. ~~The pipeline in~~ Fig. 9.1 shows ~~QKDN-opt ML pipeline node positions wherever the nodes are hosted and~~ the general operational procedures of QKDNml~~of ML-enabled QKDN performance improvement in different layers for QKDN~~.



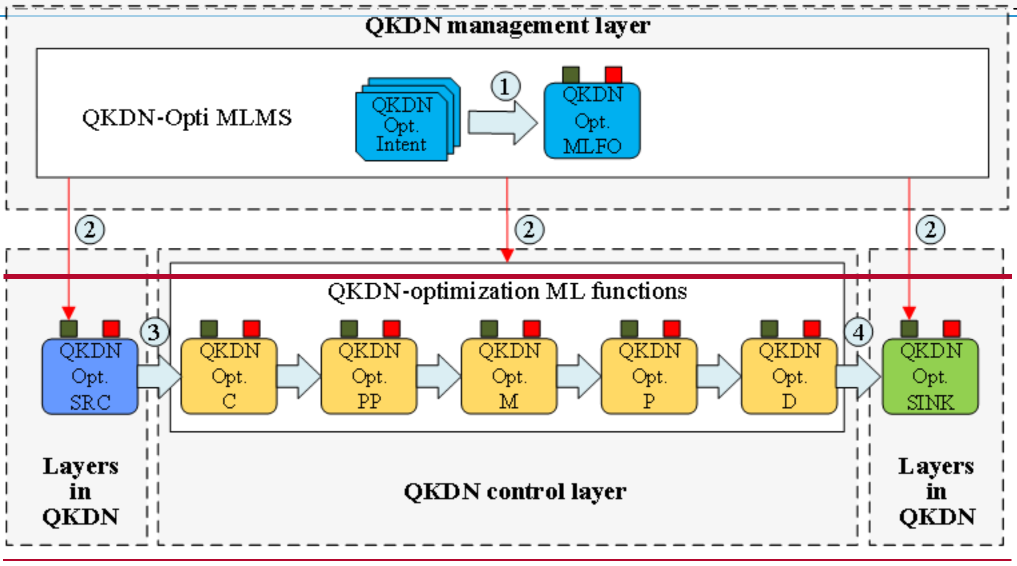


Fig. 9.1. General operational procedures of QKDNml- ~~of a QKDN-optimization ML pipeline~~

1) The QKDN performance optimization requirements are translated to the QKDN-opt intent, which is a declarative description used to specify- the QKDN optimization~~an ML application in QKDN~~. The QKDN-opt intent is input to the QKDN-opt MLFO for technology-specific implementation.

2) The QKDN-opt MLFO manages and orchestrates the nodes of QKDN-opt ML pipelines based on the QKDN intent or dynamic network conditions. The MLFO selects ML models based on the network capabilities and constraints of the QKDN-opt ML pipeline. The QKDN-opt ML pipeline functionalities are placed and chained as depicted in Fig. 9.1.

3) The QKDN-opt SRCs have different types. For different QKDN-opt intents, QKDN-opt SRCs are instantiated by different components in QKDN layers. The QKDN-opt SRCs collect the needed QKDN data from the corresponding layers of QKDN. The QKDN-opt SRCs report the collected QKDN data to the QKDN-opt ML functions.

NOTE 1 - The QKDN-opt C transfers the collected data to the QKDN-opt PP.

NOTE 2 - QKDN-opt PP cleans the collected QKDN data by removing noisy data and transforms the cleaned data into a unified data format. The pre-processed QKDN data is transferred to QKDN-opt M.

NOTE 3 - The QKDN-opt intent is performed by the deployed QKDN-opt M selected by the MLFO. The outputs of the ML model are transferred to the QKDN-opt P.

NOTE 4 - Given ML output of QKDN-opt M, a QKDN-opt policy decision can be made by the QKDN-opt P. The policy decision results are transferred to the QKDN-opt D.

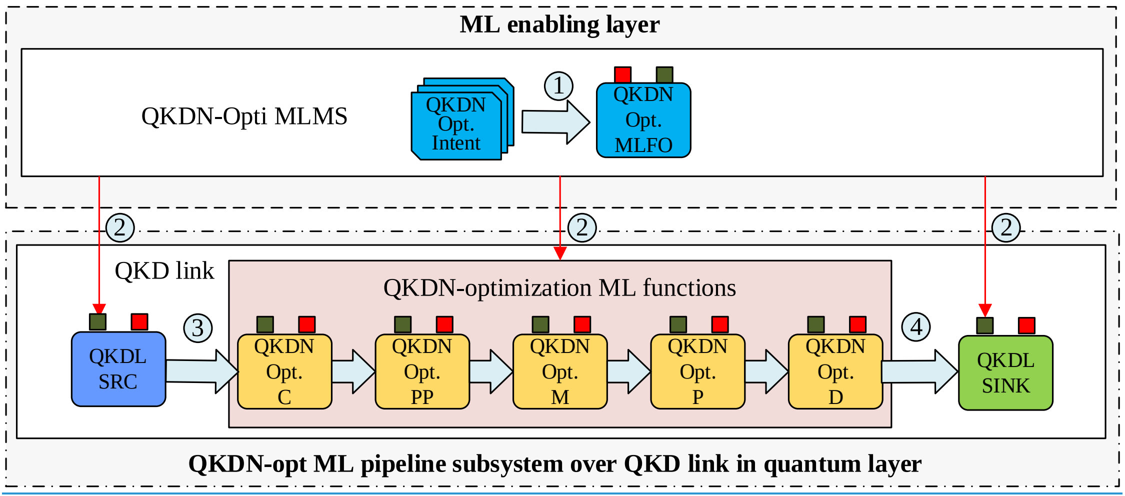
4) The QKDN-opt D distributes the QKDN performance optimization policies to the QKDN-opt SINKs instantiated by the components in QKDN layers corresponding to the QKDN-opt intent.

Appendix I

Use case of ~~a ML pipeline~~the operational procedures for ML-enabled quantum channel performance prediction

(This appendix does not form an integral part of this Recommendation.)

During the QKD process, the noise in the quantum channel will reduce the quality of the quantum channel and cause low key rate.~~ Fig I.1 shows t~~The use case of~~ a ML pipeline for~~ ML-enabled quantum channel performance prediction ([b-ITU-T Y-Suppl.70]) is shown. Firstly, the quantum channel related data and the corresponding quantum channel performance are collected through quantum channel measurement for ML model training and testing. Then, with the trained ML model, the quantum channel performance can be predicted based on the current input quantum channel related data. Lastly, according to the predicted channel performance, feedback and adjustment can be finished in advance to improve the channel environment and reduce unnecessary loss caused by key rate decreases. The detailed operational procedures are ~~as follows~~in Fig. I.1.



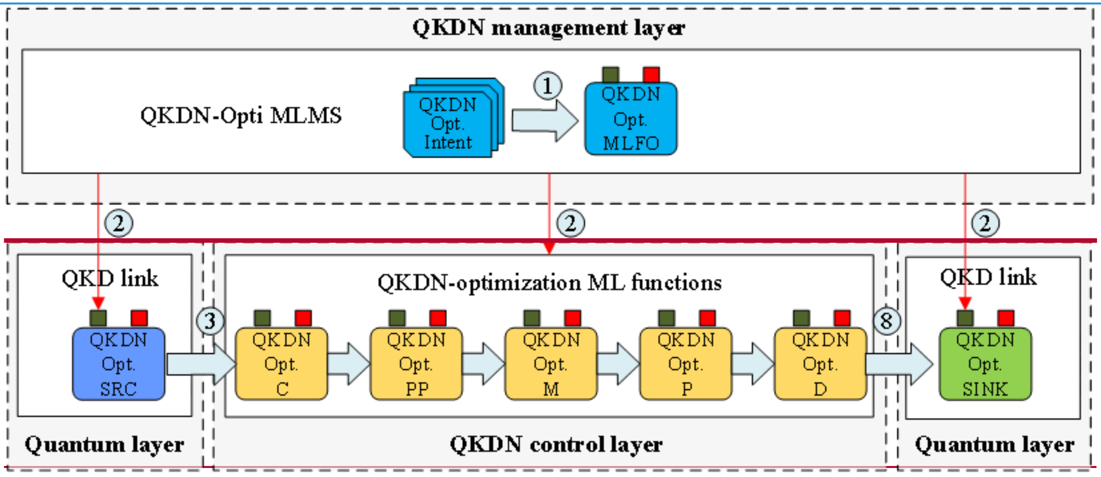


Fig. I.1. ~~use case of~~ operational procedures~~a ML pipeline~~ for ML-based quantum channel performance prediction

1) The QKDN-opt intent is used to specify the quantum channel performance prediction, and then is input to the QKDN-opt MLFO. The quantum channel performance prediction is to predict the future quantum channel performance under different channel noise environments, so that measures can be taken in advance based on the predictions to improve the channel environment and make the quantum channel in an optimal performance state.

2) QKDN-opt MLFO manages and orchestrates the ML pipeline subsystems to realize ML-based quantum channel performance prediction. The QKDN-opt MLFO connects ML nodes together

to form an ML pipeline. The data that manages and orchestrates the nodes of the QKDN-opt ML pipelines is reported to all nodes related to quantum performance prediction, including QKDN-opt SRC, C, PP, M, P, D and SINK.

3) The QKDN-opt SRC collects the quantum channel parameters of QKD links in quantum layer and then reports the data to the QKDN-opt ML functions.

NOTE 1 - The QKDN-opt C transfers the collected data to the QKDN-opt PP, which includes the quantum-channel-performance-related parameters, such as QBER of quantum channel, the SPD photon detection output counter and code formation rates under different noise environments.

NOTE 2 - QKDN-opt PP cleans the collected QKDN data by removing noisy data and transforms the cleaned data into a unified data format. The pre-processed QKDN data is transferred to QKDN-opt M after intelligent analysis.

NOTE 3 - Quantum channel performance prediction is performed by the ML models in the QKDN-opt M. The outputs of the ML model are the predicted quantum channel performance values, which are transferred to the QKDN-opt P.

NOTE 4 - Given the predicted quantum channel performance, a QKDN-opt policy decision is made by the QKDN-opt P to minimize impacts when the output of ML is applied to a live network. The policy decision results are transferred to the QKDN-opt D.

4) The QKDN-opt D distributes the predicted quantum performances to the QKD~~N-opt~~L SINK instantiated by QKD links in the quantum layer.

Bibliography

[b-ITU-T Y-Suppl.70]

Supplement 70 to ITU-T Y.3800-series Recommendations (2021), *Quantum key distribution networks* - *Applications of machine learning*.

____