# Decentralized Key Management and Service in Quantum Key Distribution Networks: An Experimental Implementation

Jian Li, *Senior Member, IEEE,* Peng Zheng, Zhonghui Li, *Member, IEEE,*
Yuqi Yang, Nenghai Yu, Qibin Sun, *Fellow, IEEE,* Jun Lu

***Abstract*—In recent years, multi-hop Quantum Key Distribution (QKD) network has been proven as a promising solution through rigorous practices to provide end-to-end key exchange service for arbitrary communication parties. However, existing decentralized solutions still face critical challenges including consistency and fairness that stem from storable nature of quantum key material. Thus, in this paper, we first devise a Key Management and Service (KM&S) framework for decentralized multi-hop QKD networks, which provides functional decoupling and pipeline processing to guarantee flexibility and compatibility for practical implementation. After that, to address consistency and fairness challenges during end-to-end key exchange service, we focus on two aspects including local key management and end-to-end congestion control, and respectively propose an elastic key supply rate control scheme named AUTO and a Capacity Probing-driven Backpressure Flow Control (CP-BFC) scheme. Furthermore, we construct an experiment platform equipped with realistic QKD devices based on China metropolitan QKD network topology to implement the proposed framework and schemes, and conduct extensive experiments compared to representative schemes in existing studies. The experimental results show that AUTO&CP-BFC significantly outperforms representative schemes in terms of consistency and fairness.**

***Index Terms*—Key management and service, quantum key distribution networks, key supply rate control, congestion control, experimental implementation**

## I. Introduction

As a novel technology that can achieve information-theoretical security even in the era of quantum computing [1, 2, 3], Quantum Key Distribution (QKD) is expected to have a significant impact on current cryptographic systems. Based on numerous studies from both academia and industry, e.g., decoy-state protocol design [4] and security analysis [5], QKD technology experiences rapid development for decades from initial proof-of-concept demonstrations in the laboratory [6] to long-haul trunk QKD networks with hundreds of nodes over optical links spanning thousands of kilometers [7, 8, 9, 10]. In the near future, it can be foreseen that a large-scale QKD network can be widely deployed to provide ubiquitous security service for various applications in terms of business, finance, and government.

Up to now, trusted relay-based interconnection approach has emerged as the mainstream for the extension from point-to-point QKD communication to multi-hop QKD networks, and thus enables the provision of end-to-end key exchange service between arbitrary communication parties [11]. Inherently, trusted relay-based QKD networks still relies on physical and logical infrastructure of conventional IP networks, e.g., the overlay network built on TCP/IP protocol stack in SECOQC QKD network [8], an evolving development architecture integrating QKD networks and IP networks is recognized as an essential development path. In this context, a decentralized QKD network configuration, which provides security service with the required efficiency, stability, and fairness akin to conventional IP networks [12, 13], is expected as one promising solution and attracts worldwide attention for its future implementation.

The multi-hop QKD network operating in a decentralized manner has been investigated over a decade from the perspective of both practical implementations [8] and theoretical studies [14, 15, 16]. Unfortunately, building an efficient, stable, and fair QKD network still faces critical challenges stem from storable nature of quantum key material: (i) *Inaccurate capacity estimation*. Instead of constant and continuous link capacity model in conventional IP networks, a discrete and dynamic key generation process is ubiquitous for realistic QKD devices, leading to inappropriate decision-making and performance degradation. (ii) *Inconsistent service performance*. Most of existing studies adopt an on-demand key supply strategy where all key material on QKD nodes is available to arbitrary user. Thus, each user can easily acquire performance improvement from abundant key generation process and potentially result in "cliff effect" on QKD node due to excessive consumption of storable key material, which can lead to dramatically inconsistent performance behavior. (iii) *Unfair resource usage*. Greedy transport control strategy in conventional IP networks, such as greedy loss-based congestion control [17], tends to exhaust available capacity along the routing path, which can lead to fairness problem for multi-user scenarios due to "first-arrival advantage". Thus, an unbridgeable gap between identical QKD network implementation and current decentralized QKD network configuration remains intractable and poses urgent challenges to be resolved.

To tackle these urgent but unresolved challenges and facilitate the pace from classical cryptography based IP networks to quantum secure communication networks, in this paper, instead of well-studied routing approaches, two rarely considered aspects including local key management and end-to-end congestion control are highlighted and investigated. At first,

J. Li, P. Zheng, Z. Li, Y. Yang, N. Yu, Q. Sun and J. Lu are with the School of Cyber Science and Technology, University of Science and Technology of China, Hefei 230027, China.

Corresponding Author: Zhonghui Li (leestone@ustc.edu.cn).

we devise a Key Management and Service (KM&S) framework to implement decentralized modules. KM&S not only provides functional decoupling to support flexible deployment of key management and encryption suite, but also achieves standard pipeline processing to guarantee compatibility with conventional IP networks. Second, we propose an elastic key supply rate control scheme named AUTO, which can periodically adjust key supply rate through queuing optimization, to address both capacity estimation challenge from on-demand key supply strategy and consistency challenge triggered by excessive consumption of key material. Third, inspired by existing studies in classical networks [18], we propose a Capacity Probing-driven Backpressure Flow Control (CP-BFC) scheme to avoid aggressive traffic entry and thus guarantee fairness among multiple users. At last, we construct an experiment platform equipped with realistic QKD devices based on China metropolitan QKD network topology, and implement the proposed KM&S framework and AUTO&CP-BFC scheme through an overlaying IP network. In addition to the verification and performance comparisons, experimental results also provide theoretical guidance for future deployment of decentralized multi-hop QKD networks.

The main contributions of this paper can be summarized as follows:

- To achieve flexible configuration and compatible implementation, we devise a KM&S framework for decentralized multi-hop QKD networks. KM&S adopts decoupled functional module design and pipeline processing, and thus supports various management and control schemes and both overlaying and underlying implementation in TCP/IP protocol stack.
- We propose an elastic key supply rate control scheme through queuing optimization, named AUTO, to achieve local key management. On the one hand, AUTO dynamically adjusts the real-time key supply rate according to local status of key storage and key request traffic, and guarantees both consistency for each user's requests and elasticity for burst traffic from multiple users. On the other hand, AUTO can provide periodically stable capacity on each node to achieve accurate capacity estimation.
- We propose a Capacity Probing-driven Backpressure Flow Control (CP-BFC) scheme. Based on periodically stable capacity determined by AUTO, CP-BFC maintains per-flow queue and strictly limits incoming key request messages from each user, thereby preventing aggressive traffic entry through per-flow monitoring and eliminating congestion loss caused by excessive traffic in the queue. Combined with fair scheduling strategy, CP-BFC can guarantee fair resource usage among multiple users.
- We construct an experiment platform equipped with realistic QKD devices to implement KM&S framework and AUTO&CP-BFC scheme, and conduct extensive experiments compared to representative schemes in existing studies. The implementation verifies the flexibility and compatibility of KM&S framework, and experiment results show that the proposed AUTO&CP-BFC outperforms existing schemes in terms of consistency and fairness under both single and multiple user scenario.

The rest of the paper is organized as follows. Section II mainly introduces the related work and motivation, and Section III elaborates the specific components of decentralized KM&S framework. After that, a local key management scheme, i.e., elastic key supply rate control scheme is introduced in Section IV, and then a congestion control scheme, i.e., CP-BFC scheme is introduced in Section V. In Section VI, the specific experimental implementation and results are presented. Finally, our work is concluded in Section VII.

*Concept Notation:* In this paper, port refers to egress port by default, which consumes key material from master key storage, and upstream port represents the certain port on previous node that sends key request traffic to current node along the routing path $\mathcal{P}(f)$ of $f$-th flow.

*Symbol Notation:* Boldface letters denote mathematical set, $|\mathbf{a}|$ denotes the total element number in a set $\mathbf{a}$, and $[\cdot]^+ \triangleq \max\{0, \cdot\}$.

## II. Related Work and Motivation

### A. Related Work

A trusted relay-based multi-hop QKD network can extend information-theoretical secure key generated from point-to-point QKD process to end-to-end key exchange service. In recent years, representative QKD network projects using trusted relay approach have been successively developed around the world, which demonstrate the significant development in terms of higher key generation rate, longer distances, and larger scale. To accelerate the development speed for widespread applications in near future, the quality of end-to-end key exchange service from user's perspective should be considered and optimized. Although the end-to-end key exchange process is similar to end-to-end packet forwarding process in conventional IP networks, a straightforward application over full-fledged TCP/IP protocol stack still exists numerous challenges, e.g., congestion loss, route decision, and key management.

To address these challenges, previously deployed QKD network projects have put great efforts in practice [11, Table 6]. For congestion loss, SECOQC QKD network [8] introduces new techniques called QKD-TL to prevent network congestion, it tries to prevent congestion condition by reacting proactively on the basis of signaling mechanism triggered by key store shortages similar to Explicit Congestion Notification (ECN). For route decision, DARPA QKD network [7] adopts a modification of well-known Open Shortest Path First (OSPF) routing protocol by using available key resources on links as estimation criterion. For key storage management, China QKD networks [10, 19, 20] utilize centralized control station to collect the running status of key storage capacity, and adopt best-effort key supply strategy under One-Time-Pad (OTP) encryption and variable refresh rate strategy under symmetric encryption algorithms. To further provide QoS guarantees, China QKD networks also apply Resource Reservation Protocol (RSVP) as a transport-level signaling protocol [21] used with variety of QoS control services.

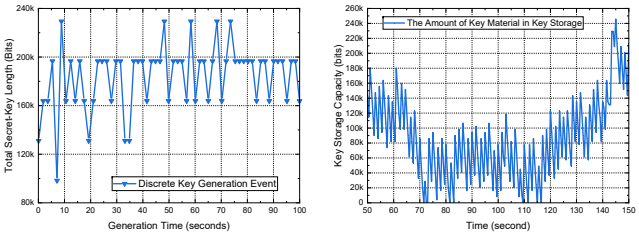
In addition to practical solutions proposed in QKD network projects, there are also many theoretical investigations

to explore every potentials. At early stage, existing studies concentrates on the precious resource due to the limited key generation rate is mainly considered in existing studies. Yang [22] *et al.* firstly considered the impact of key resources to the routing design, and proposed a secret-key-aware routing method. Although the proposed routing method can achieve load balance to avoid key shortage for relay nodes and significantly improve key exchange success rate compared to traditional shortest path first routing design, inherently, it only adopts a threshold-based searching metric and still does not deeply investigate the relationship between local key supply and overall network performance. To guarantee the QoS during end-to-end key exchange process, various per-flow-based resource reservation protocols are proposed inspired by integrated services in Internet [23], e.g., QSIP [24], Q3P [25], RSVP [21]. Those solutions can reserve key material along the routing path in advance to ensure resource availability for single flow and are compatible with IP network over overlaying implementation. After that, Mehic [14] *et al.* proposed a novel quality-of-service model based routing protocol inspired from ad-hoc networks to achieve high-level scalability and minimization of quantum key consumption. Although they provide both integrated service with QoS guarantees, which is achieved by RSVP to reserve per-flow resource before key exchange process, and differentiated service without QoS guarantees. However, the specific routing design considers a continuous key generation model and a threshold-based key management model, and thus a stable performance for each user can hardly be guaranteed and packet loss likely occurs. In 2022, Zhou [15] *et al.* noticed the critical issue triggered by limited quantum key resources, and thus designed a key management and data scheduling scheme, which can optimize the utility of data transmission and offer a stable data queue. However, the adopted Lyapunov optimization technique not only places an artificial constraint on the key storage, but also requires centralized information collection and decision, and thus the implementation scenario is quite limited. Recently, Akhtar *et al.* [16] further proposed a secure and provably routing policy that supports multicast and anycast traffic. Although it further supports multicast and anycast traffic, a centralized information collection is still necessary.

Unfortunately, the existing studies still lacks consideration of performance consistency and multi-user fairness from the perspective of local key management and end-to-end congestion control. In this paper, instead of well-studied routing approach and centralized solutions, two rarely considered aspects including local key management and end-to-end congestion control are highlighted and investigated. In the following, we further highlight the motivation of our work.

### B. Motivation

To exhibit the unique challenges of decentralized multi-hop QKD network, which exist but are normally neglected in existing studies, we conduct preliminary experiments based on a dummy network topology equipped with realistic QKD devices, and adjacent QKD devices (QKD-PHA1250, QuantumCTek Co., Ltd. [27]) are connected through 10km optical fiber. The observations are discussed as follows.




(a) Key generation event (b) Real-time capacity of key storage

Fig. 1. **Discrete and Dynamic Key Generation Process:** Illustration of a simple end-to-end key exchange process in single user scenario with a constant key request sending rate (60kbps). The topology is same as Fig. 4, and the specific generated secret-keys from a QKD device connected to relay node and real-time key storage capacity on relay node are depicted, respectively. All key generation event is collected from QKD devices and the corresponding open-source dataset is available in [26].

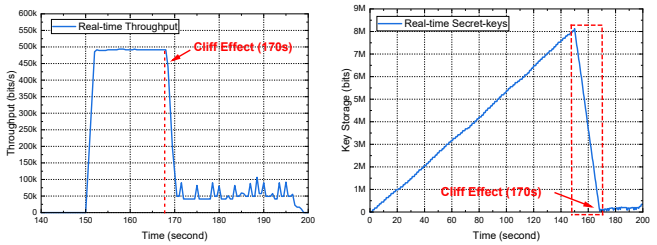
***1. Inaccurate Capacity Estimation:*** This challenge is triggered by the nature of discrete and dynamic key generation process. In multi-hop QKD networks, a QKD node consists of QKD devices and interconnected Key Management Terminal (KMT) devices, key material is generated between point-to-point QKD devices and gets pushed to the key storage in KMT through private interface. The observation in Fig. 1(a) depicts the discrete and dynamic key generation process of a typical QKD device. Each data point presents a key generation event[1], which means a QKD device outputs a batch of secret-keys to key management terminal through QKD interface. Thus, it should be highlighted that a simple and continuous model can hardly describe the key generation process in practice. Fig. 1(b) exemplifies the real-time capacity of key storage while serving key requests with a constant sending rate (60kbps), but the capacity still lacks stability due to the discrete and dynamic key generation process. In conventional IP networks, link capacity estimation is critical in transport decision-making such as congestion control, inaccurate estimation results can lead to severe congestion problem. However, most of existing studies consider continuous key generation model [14, 15, 16, 22] and constant key generation rate [14, 15, 22]. Thus, the performance of these proposed schemes might experience significant performance degradation in practical implementation. According to the observations, we provide the following remarks.

**Remark 1.** *The key generation rate $\lambda(t)$ between arbitrary adjacent nodes provided by QKD devices follows stochastic process, i.e., $\lambda(t) \sim f(\mathbf{x}), \forall t$.*

**Remark 2.** *The key generation event received by KMT from QKD devices follows a discrete arrival process. Specifically, the discrete key generation time $t^i$ and the amount of generated key material $\Lambda(t^i) = \lambda(t^i) \cdot (t^i - t^{i-1})$ at $t^i$ is variable.*

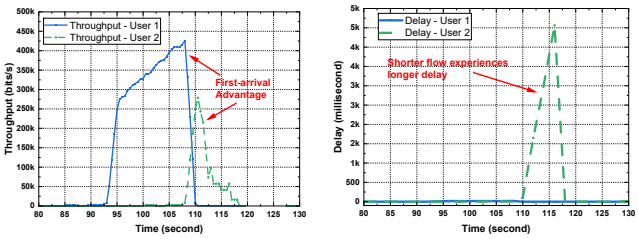
***2. Inconsistent Service Performance:*** This challenge arises from widely-deployed on-demand key supply strategy used by relay nodes. Most of existing studies, e.g., most of deployed such as Q3P in SECOQC project [8] and theoretical

[1]This phenomenon is discussed in [8, Chapter 5.1.1], and the specific key generation process can also be significantly influenced by the operating environment such as temperature according to our observations.




(a) Throughput vs. time (b) Length of key storage vs. time

Fig. 2. **Cliff Effect:** Key exchange performance in terms of throughput and real-time status of key storage (w/ cumulative secret-keys in advance for 150 seconds).




(a) Throughput vs. time (b) End-to-end delay vs. time

Fig. 3. **First-arrival Advantage:** Key exchange performance in terms of throughput and delay under two user scenario (w/ cumulative secret-keys in advance for 85 seconds).

approaches in [14, 16, 22], aim to provide end-to-end key exchange service for users' key requests in an on-demand manner, and thus accumulated key material from abundant key generation process in key storage can be excessively consumed by each user, which can easily result in cliff effect. Experimental results, as illustrated in Fig. 2, depict that an accelerated sending rate of key request messages during 150s to 168s induces a cliff effect due to the exhaustion of key material after 168s. On the one hand, cliff effect event indicates that the relay node loses elasticity to absorb burst key requests. On the other hand, dramatical capacity changing behavior poses consistent challenges for traditional transmission control mechanism, e.g., load-balanced routing and congestion control in TCP. In this context, a stable yet elastic key supply rate is preferred.

***3. Unfair Resource Usage:*** This challenge stems from excessive traffic entry in congestion control mechanism used by the sender. In conventional IP networks, TCP is considered as a widespread closed-loop control mechanism, which dynamically adjusts congestion window according to the real-time network status (e.g., packet loss is frequently servers as the congestion signal). Due to the memoryless characteristic of port capacity, each TCP sender can continuously increases the congestion window as long as the network status remains stable (e.g., no packet loss happens) [28]. However, in QKD networks, the storable nature of key material must be taken into account in congestion control mechanism, otherwise, arbitrary TCP sender can easily inject excessive traffic as long as key material remains available, which momentarily boosts throughput but rapidly exhausts key material in key storage, resulting in prolonged periods of poor performance. As illustrated in Fig. 3, two users sequentially send key request traffic and a round-robin scheduling strategy is deployed. User 1, sending key requests traffic at first, acquires sufficient key material and achieves maximum throughput, but diminishing the resource in key storage. As a result, user 2 can hardly achieve comparable performance due to resource shortage, even though user 2 (5 Mbits) only sends half shorter flow than user 1. In this case, a fair usage of precious key material can hardly be guaranteed.

### C. Design Goals

To address the challenges described in Sec. II-B, three design goals for a decentralized multi-hop QKD network are

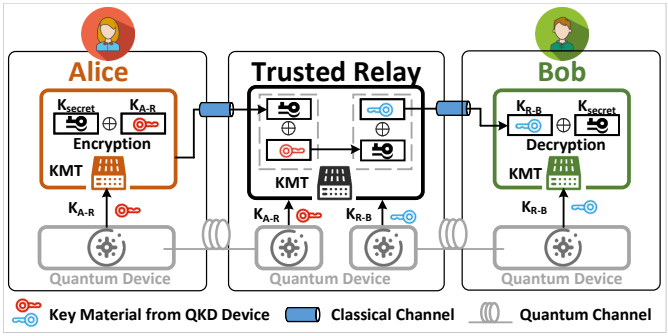



Fig. 4. Illustration of typical end-to-end key exchange process via OTP encryption in multi-hop QKD networks.

expected as follows.

1. **Accurate Capacity Estimation:** Each node in the network can acquire an accurate capacity estimation and determine corresponding transport decisions. As a result, no message loss is expected during end-to-end key exchange service.
2. **Consistent Service Performance:** For each user's key requests within different time period, instead of on-demand key supply strategy, QKD network can offer consistent service performance over time in terms of throughput and end-to-end latency through local key supply rate control.
3. **Fairness Guarantee:** For multi-user competition, instead of exhausting key storage and providing best-effort performance, QKD network can guarantee fairness among multiple users and avoid *unfair resource usage* from aggressive users.

## III. Key Management & Service Framework

In this section, we first present the present the details of end-to-end key exchange service, and then introduce the specific components of decentralized KM&S framework including Key Request Management Module (KRMM), Key Supply Module (KSM), and Encryption Module (EM), respectively. In the following, detailed implementations of each module are introduced.

### A. End-to-end Key Exchange Service

In trusted relay-based multi-hop QKD networks, adjacent QKD devices can generate bit strings through point-to-point QKD protocols, e.g., decoy BB84 [4] and B91, and then put

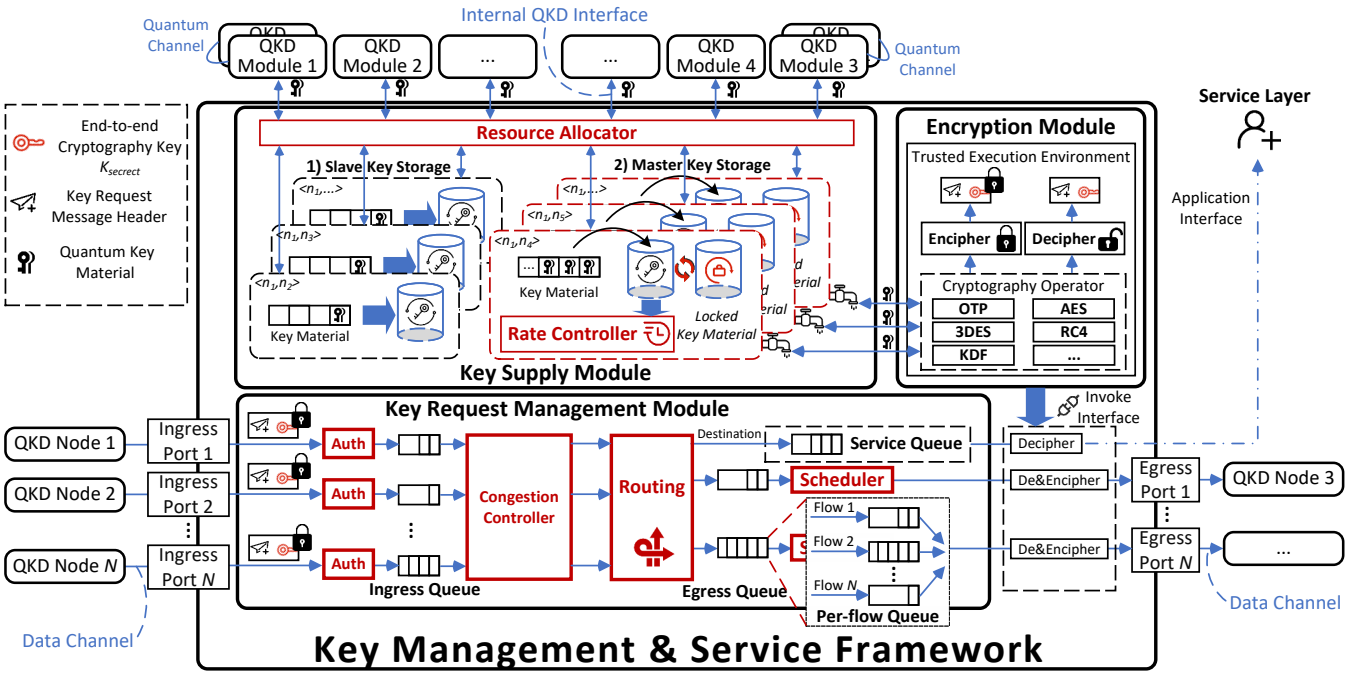



Fig. 5. **Key Management** & **Service Framework:** Illustration of the specific components including KRMM, KSM, and EM.

them into key storage in classical memory as quantum key material. Once the service extends to remote nodes, a key exchange process through multi-hop relaying is required. As shown in Fig. 4, the specific end-to-end key exchange workflow between two remote nodes can be described as follows. Alice sends the key request message to the relay node via OTP encryption by default. Relay node decrypts the message by taking XOR operation with shared secret-key $K_{A-R}$ at first, and then encrypts the message with shared secret-key $K_{R-B}$. Finally, Bob obtains an end-to-end cryptography key $K_{secret}$ between Alice and Bob over the classical channel after decryption. In this paper, we define a sequence of key requests messages from the same source-destination pair through key exchange process as the traffic flow.

For practical implementation, there are some necessary functions that determines the Quality of Service (QoS), e.g., key management, routing control, authentication control described in ITU-T [**?** ] and IETF [29], but lacks standardized solutions yet. Thus, in the following, we propose a decentralized KM&S framework for practical implementation of QKD networks.

### *B. Key Request Management Module*

KRMM provides a standard processing pipeline for all key request messages from users, including message authentication, route decision, queue management and scheduling. This kind of design can guarantee compatibility to classical Internet and thus support both overlaying and underlying implementation in practice. In Sec. VI, we conduct extensive experiments through overlaying implementation on classical switches.

**Message Format:** Currently, the standardization of QKD signaling message format has no final conclusion, existing studies have been working on such practical issue and provide the specific encapsulation format for key request message. In this paper, we refer to the encapsulation in [11, Sec. 3], and adopt AIT R10 QKD header format, which is used in the SECOQC project [8, 30], as our key request message format. For the convenience of global key management, a unit secret-key is treated as a variable-length value $Length$, whose length can be adjusted based on the application requirement. In the QKD network, $Length$ bits keys are globally applied as a unit secret-key for key storage and key request. Each key request message consumes unit key material for cryptographic protection.

**Message Authentication:** To protect the security of key request messages during the communication over classical channel, all traffic should be encrypted, authenticated, and integrity-checked [11] in KRMM. As shown in Fig. 5, once an incoming message is received from ingress port, KRMM verifies its authentication tag at first to check its validity and integrity. If the message is successfully verified, KRMM continues the processing pipeline and add another authentication tag to the message executed by EM after scheduling operations; otherwise, it should be discarded.

**Congestion Controller:** Congestion control modulates traffic entry into the network to avoid congestion collapse resulting from overwhelming traffic. Thus, congestion controller is a critical and essential component which prevents users sending overwhelming traffic. In classical networks, TCP congestion control is a representative mechanism that works in transport layer and has received significate recognition. In multi-hop QKD networks, it is challenging to directly deploy TCP-like congestion controller due to the unique characteristic of quantum key material. For example, packet loss event is considered as the indicator of a congestion event in TCP

Reno and thus reduces congestion window, but such packet loss-based reaction is lagging since packet loss in QKD networks represents the exhaustion of key material on the relay node. Although some existing QKD projects adopt TCP-like congestion controller such as QKD-Transport Layer in SECOQC [8], it also requires additional modifications to make it work. In KRMM, the congestion controller maintains the status of each traffic flow (e.g., historic round-trip time and acknowledgements) and determines the sending rate of each traffic flow. In Sec. V, we propose a novel congestion control that leverages per-flow queues and local key management to achieve fair resource usage among multiple users.

**Route Decision:** As one of the critical components that impacts the networking performance, routing algorithm determines the global route path during end-to-end key exchange process. In KRMM, each key request message is forwarded to corresponding egress port according to the route decision. In multi-hop QKD networks, considering the unique characteristic of quantum key material, existing studies also attempt to design efficient routing algorithms, we list several representative routing designs as follows:

- *Shortest Path First-based Routing:* This kind of routing design stems from conventional IP networks, such as Dijsktra algorithm [31] and widely deployed OSPF routing protocol [32], which provides a simple but efficient solution. It should be noticed that OSPF is also adopted in practical QKD network projects, e.g., SECOQC QKD network.
- *Key-resource-aware Routing:* Some recent studies [14, 15, 16, 22, 33] further take the amount of key material into account, and devise adaptive routing algorithms for multi-hop QKD networks. This kind of routing design exploits real-time resource information, e.g., key generation rate and current residual key quantity, to search for the optimal routing path.
- *Others:* There are also other routing designs for various optimization goals, e.g., multi-path and stochastic routing algorithms [34, 35] for security protection. KRMM supports customized routing algorithms.

**Key Request Scheduler:** During the end-to-end key exchange service as described in Sec. III-A, a scheduler is necessary to allocate port capacity for multiple users' requests. As shown in Fig. 5, KRMM maintains per-flow queues in the buffer. The processing sequence of all key request messages stored in per-flow queues are decided by the scheduler, which can be set as manually according to the practical requirement. Once the destination is the node itself, the corresponding messages enqueues a special queue called service queue waiting to be decapsulated and upload to the service layer through application interface. We list some optional schedulers as follows.

- *Round-Robin Scheduler*: Round-robin scheduler processes messages in each per-flow queue one-by-one. Its simplicity makes it become the most common scheduler [36], which can achieve inter-flow fair scheduling, and achieve good performance in general.
- *Prioritized Round-Robin Scheduler*: This kind of scheduler processes messages in each per-flow buffer based on its weight factors. According to differentiated priority requirement such as QoS, flows belong to important applications or users are assigned to higher weight factors.
- *Others*: Since each message consumes the constant key material from key storage, a simple scheduler can handle most of the requirement in multi-hop QKD networks. Meanwhile, KRMM also supports other customized schedulers.

According to the illustration in Fig. 5, each key request message is processed by congestion controller, routing algorithm, and scheduler in sequence. It should be noted that the specific algorithms adopted in each component are compatible as long as they follow the workflow in KRMM. For example, the proposed congestion control scheme in Sec. V is compatible to shortest path first-based routing algorithms and also aforementioned scheduling algorithms. Unfortunately, some routing designs such as [15, 16] can hardly be deployed in decentralized KM&S framework since it requires centralized controller to achieve information collection.

### C. Key Supply Module

**Key Synchronization:** Inherently, QKD devices adopt point-to-point communications to generate secret-keys (i.e., key material) for both adjacent nodes' usage. Since key requests in QKD networks require a directional routing process, each QKD node must allocate generated key material from key storage to encryption or decryption, and thus the key synchronization mechanism becomes necessary. Specifically, there are two critical problems behind synchronization between key managers, i.e., *conflict avoidance for key usage* and *consensus for key supplying*. The corresponding solutions can be described as follows.

To address the problem regrading *conflict avoidance for key usage*, i.e., how to avoid conflicts between key managers while using secret-keys, KSM provides two types of key storage (similar to the in/out buffers in [8]), i.e., master key storage for key encryption, and slave key storage for key decryption. Once key material is generated from QKD devices, they are allocated to these key storage. For each egress port on QKD node, all key material in master key storage can be independently determined for encryption without notifications with adjacent node. Meanwhile, all key material in slave key storage can be passively accessed according to the authentication tag and encryption index in key request messages. The specific allocation method can be determined according to the statistic load condition. For stable network environment that maintains consistent and stable traffic patterns, a simple solution such as equal allocation is sufficient. For highly dynamic network environment that traffic patterns evolves over time, a sophisticated solution, e.g., dynamic allocation based on statistic key storage status, is preferred.

To address the problem regrading *consensus for key supplying*, i.e., how to reach the consensus of key supplying to applications, KSM provides an acknowledgement-based key reservation mechanism. After end-to-end key exchange process, end-to-end cryptography keys can be stored between

remote key managers. However, due to the uncertainty of key exchange process (e.g., delay and success signal), a synchronization mechanism is required to reach consensus about the results of key exchange process, and thus the remote key managers can provide symmetric secret-keys to the applications and ensure that two communication parities can encryption and decryption. For the source node of an arbitrary key request message, an end-to-end cryptography key (i.e., $K_{secrect}$ in Fig. 4) in the key storage on source node is reserved temporarily. Once the key exchange process is successfully finished, the destination node sends an acknowledgement message to the source node, and then source node and destination reach a consensus. Thus, the reserved end-to-end cryptography key $K_{secrect}$ can be released for key supplying service, which is also a conflict-free process due to the master-slave key storage design.

**Key Supply Rate Controller:** The maximum capacity of a port on QKD node is decided by the amount of key material in master key storage. Most of existing studies consider an on-demand key supply strategy. As described in Sec. II-B, on the one hand, a discrete and dynamic key generation process can lead to inaccurate capacity estimation; on the other hand, on-demand key supply strategy can lead to consistency issue. Thus, an active key management design becomes necessary. In the next section, we propose an elastic key supply rate control scheme to dynamically adjust key supply rate at egress port by consideration of both key request traffic and key generation status.

### D. Encryption Module

**Cryptography Algorithms:** To achieve information-theoretical security, OTP encryption is applied by default in multi-hop QKD networks. However, due to the limited key generation rate for point-to-point QKD, symmetric encryption algorithms such as advanced encryption standard [11, 37] and derivation based algorithms [38] are also considered as alternative approaches.

**Trusted Execution Environment:** Since end-to-end cryptography key $K_{secret}$ described in Sec. III-A contained in key request messages become plaintext after decryption on relay nodes, a promising security protection is necessary especially for future integrating QKD networks and IP networks. A simple but effective solution is that, EM utilizes a standalone hardware called trusted execution environment, e.g., ARM TrustZone and Intel SGX on universal devices [39], to provide cryptography operators and avoid plaintext leakage threat. All cryptography operations can be executed by invoking EM's external interface, KRMM and KSM provide key request messages and assigned key material, respectively.

## IV. Elastic Key Supply Rate Control Scheme

In this section, we propose an elastic key supply rate control scheme to address *inaccurate capacity estimation* and *inconsistent service performance* challenges through periodical adjustment and queuing optimization, respectively. At first, we mathematically study the long-term key supply problem and derive several principles for steady-state guarantees. After that, under the guidance of these principles, we formulate an optimization for real-time key supply rate control and devise an effective algorithm called AUTO to automatically determine real-time key supply rate at each port on QKD nodes.

TABLE I
SYMBOLS USED IN THE PAPER

| Symbol | Notation |
|---|---|
| $W$ | Elastic window size in the QKD network. |
| $T$ | Update interval for KSM periodically updating real-time key supply rate. |
| $\mathbf{t}_n^m$ | History key generation time set at $m$-th port on $n$-th node, and $i$-th key generation time satisfies $t_n^{m,i} \in \mathbf{t}_n^m$. |
| $a_n^m(t+T)$ | Key request arrival rate at $m$-th port on $n$-th node within time slot $[t, t+T]$. |
| $A_n^m(t+T)$ | The number of messages arrived at the queue at $m$-th port on $n$-th node within time slot $[t, t+T]$, and $A_n^m(t+T) = a_n^m(t+T) \cdot T$ is satisfied. |
| $s_n^m(t+T)$ | Key supply rate at $m$-th port on $n$-th node within time slot $[t, t+T]$. |
| $S_n^m(t+T)$ | The amount of key material supplied for key exchange process at $m$-th port on $n$-th node within time slot $[t, t+T]$, and $S_n^m(t+T) = s_n^m(t+T) \cdot T$ is satisfied. |
| $\lambda_n^m(t)$ | Real-time key generation rate of adjacent QKD devices at time $t$ at $m$-th port on $n$-th node. |
| $\Lambda_n^m(t_n^{m,i})$ | The amount of key material generated from QKD devices at $m$-th port on $n$-th node within $[t_n^{m,i-1}, t_n^{m,i}]$, and $\Lambda_n^m(t_n^{m,i}) = \lambda_n^m(t_n^{m,i}) \cdot (t_n^{m,i} - t_n^{m,i-1})$ is satisfied. |
| $L_n^m(t)$ | Total amount of key material in key storage at time $t$ at $m$-th port on $n$-th node. |
| $Q_n^m(t)$ | Virtual queue length that records total key request messages at $m$-th port on $n$-th node at time $t$. |
| $\tilde{Q}_n^{m,f}(t)$ | Per-flow queue length for $f$-th flow at $m$-th port on $n$-th node at time $t$. |
| $\mathbf{F}_n^m$ | Key request flow set at $m$-th port on $n$-th node. |
| $\mathcal{P}(f)$ | Routing path set $\{n \mid n \in \mathcal{N}\}$ of $f$-th traffic flow. |

### A. Elastic Window Mechanism

Due to the storable nature of quantum key material, each QKD node is capable to serve an elephant traffic flow or multiple mice traffic flows that exceeds key generation rate as long as the key storage is not empty. In this paper, the magnification between real-time key supply capacity and long-term key generation capacity is called service elasticity. For example, in Fig. 2(b), the long-term key generation rate of adjacent QKD devices is 50kbps. After 150 seconds accumulation, the QKD node can supply 8Mb key material to serve an elephant traffic flow with nearly 500kbps key supply rate at 150s, which is 10-fold higher than long-term key generation rate. Thus, to describe the real-time capacity of a key storage and describe cliff effect from mathematical perspective, we

further propose a novel concept, i.e., the Degree of Elasticity (DoE) as follows.

**Definition 1.** *The DoE at $m$-th port on $n$-th node at time $t$ is defined as follows:*

$$DoE_n^m(t) = \frac{L_n^m(t) - \bar{\Lambda}_n^m(t_n^{m,i})}{\bar{\Lambda}_n^m(t_n^{m,i})}, t \in [t_n^{m,i}, t_n^{m,i+1}],$$

*where $\bar{\Lambda}_n^m(t_n^{m,i}) = \sum_{i=1}^{i}[\lambda_n^m(t_n^{m,i}) \cdot (t_n^{m,i} - t_n^{m,i-1})]/|\mathbf{t}_n^m|$ denotes the average amount of key material generated in one key generation event, and $L_n^m(t)$ indicates the maximum key supply capacity at $m$-th port on $n$-th node at time $t$.*

According to the definition, DoE measures the real-time service elasticity of a given port on the QKD node. A higher DoE represents more abundant key material in key storage, and indicates the QKD node is capable of serving more key request traffic. Otherwise, a lower DoE, especially when $DoE < 0$, only enables traffic below the long-term key generation rate to be served. Based on the definition of DoE, a cliff effect can be mathematical defined as follow.

**Definition 2.** *The cliff effect indicates that $DoE_n^m(t) < 0$ is satisfied at $m$-th port on $n$-th node at time $t$.*

As described in Sec. II-B, on-demand key supply strategy can lead to *unfair resource usage*. However, an overconservative key supply strategy can also limit the port capacity and make it difficult to absorb burst key requests from multiple users, i.e., peak-load shifting. Thus, we devise a sliding elastic window mechanism to absorb burst key request traffic but also guarantee cliff effect avoidance. The elastic window size is denoted as $W$, an elastic window provides a restricted usage of key material in key storage. We adopt a linear constraint to restrict redundant key material for burst traffic in case key storage exhausting. According to the definition of degree of elastic, the redundant key material in key storage can be calculated by $DoE_n^m(t) \cdot \Lambda_n^m(t)$, and we only tolerate exhaustion of key storage after $W$ times key generation event with key supply rate $s_n^m(t+T)$. The constraint can be written as

$$s_n^m(t+T) \leq \frac{DoE_n^m(t) \cdot \Lambda_n^m(t)}{W \cdot \bar{t}_n^{m,i}} + \bar{\lambda}_n^m(t),$$

where $\bar{t}_n^{m,i}$ and $\bar{\lambda}_n^m(t)$ represent average time interval between key generation event and key generation rate at $m$-th port on $n$-th node from history record, respectively.

### B. Key Supply Rate Control: Long-term Analysis

To provide consistent service performance, at first, we formulate a long-term key supply problem and analyze the properties that should be satisfied to maintain long-term steady state. Due to the discrete arrival property of key generation process, we consider a constant key supply rate $s(t+T)$ at each time slot $[t, t+T]$ and transform the problem as a long-term queuing problem.

As shown in Fig. 6, we consider a typical end-to-end key exchange scenario. Source node continuously sends key request messages to the destination node, and the relay node receives incoming messages along the routing path. Specifically, the

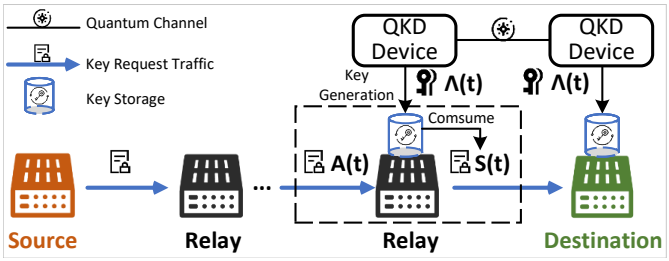



Fig. 6. A typical end-to-end key exchange process in QKD networks.

arrival rate of incoming messages within each time slot is $a(t+T)$, the constant key supply rate within each time slot is $s(t+T)$, and equivalent key generation rate within discrete key generation time slot $[t^{i-1}, t^i]$ is $\lambda(t^i)$.

To maintain a stable long-term queuing delay, the instantaneous queuing delay should converge to a constant value as follows:

$$|\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} D(t) - C| \leq \varepsilon, \tag{1}$$

where $D(t)$ denotes average queuing delay before time $t$, $C$ denotes the expected queuing delay for long-term convergence, and $\varepsilon$ is convergence gap. According to Little's law [40, 41], $Q(t) = D(t) \cdot a(t)$, Ineq. (1) can be rewritten as

$$\begin{aligned} &|\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \frac{Q(t)}{a(t)} - C| \\ &= |\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \frac{Q(t-T) - S(t) + A(t)}{a(t)} - C| \\ &= |\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \frac{Q(t-T) - S(t)}{a(t)} + T - C| \leq \varepsilon, \end{aligned} \tag{2}$$

where $A(t) = a(t) \cdot T$ and $S(t) = s(t) \cdot T$. To guarantee the convergence of the long-term queuing delay, we have the following Lemma.

**Lemma 1.** *To guarantee the convergence of the long-term queuing delay, the following constraint should be satisfied within each time slot:*

$$\bar{Q}(t-T) - (T-C-\varepsilon)\bar{a}(t) \leq \bar{S}(T) \leq \bar{Q}(t-T) + (T-C+\varepsilon)\bar{a}(t).$$

*Proof.* Please see Appendix A for detailed proof. □

Based on Lemma derived from long-term queuing analysis, we can set corresponding constraints for the guidance of real-time key supply rate control to maintain a stable service.

### C. Key Supply Rate Control: Real-time Solution

To guarantee stable queuing delay on each QKD node and provide a consistent key exchange service, for arbitrary $m$-th port on $n$-th node at time $t$, the real-time key supply rate control problem can be formulated as follows.

$$\min_{s_n^m(t+T)} \quad ||\bar{D}_n^m(t+T) - C|| \tag{3}$$

$$
\begin{aligned}
s.t. \quad & \bar{S}_n^m(t+T) \geq \bar{Q}_n^m(t) - (T - C - \varepsilon)\bar{a}_n^m(t+T), && \text{(C1)}\\
& s_n^m(t+T) \geq a_n^m(t+T), && \text{(C2)}\\
& \bar{S}_n^m(t+T) \leq \bar{Q}_n^m(t) + (T - C + \varepsilon)\bar{a}_n^m(t+T), && \text{(C3)}\\
& s_n^m(t+T) \leq \frac{DoE_n^m(t) \cdot \Lambda_n^m(t)}{W \cdot T} + \lambda_n^m(t), && \text{(C4)}\\
& s_n^m(t+T) \leq \frac{L_n^m(t) \cdot T}{t_n^{m,i+1} - t_n^{m,i}}, && \text{(C5)}\\
& Q_n^m(t) = [Q_n^m(t-T) - S_n^m(t) + A_n^m(t)]^+, && \text{(C6)}\\
& T \geq C, s_n^m(t) \in \mathbb{R}^+, && \text{(C7)}
\end{aligned}
$$

where $\bar{S}_n^m(t)$, $\bar{A}_n^m(t)$, and $\bar{D}_n^m(t)$ are historic value of $S_n^m(t)$, $A_n^m(t)$, and $D_n^m(t)$ on average, respectively. Specifically, $\bar{S}_n^m(t+T) = (\bar{S}_n^m(t) \cdot |\mathbf{t}_n^m(t)| + s_n^m(t+T) \cdot T)/(|\mathbf{t}_n^m(t)| + 1)$, $\bar{A}_n^m(t+T) = (\bar{A}_n^m(t) \cdot |\mathbf{t}_n^m(t)| + a_n^m(t+T) \cdot T)/(|\mathbf{t}_n^m(t)| + 1)$, and $\bar{S}_n^m(t)$ and $\bar{A}_n^m(t)$ come from history record. Queuing delay at time $t+T$ can be predicted by $\bar{D}_n^m(t+T) = (\bar{D}_n^m(t) \cdot |\mathbf{t}_n^m(t)| + \frac{Q_n^m(t+T)}{a_n^m(t+T)})/(|\mathbf{t}_n^m(t)| + 1)$, where $\bar{D}_n^m(t)$ is average queuing delay at time $t$ from history record, and $\frac{Q_n^m(t+T)}{a_n^m(t+T)}$ is predicted queuing delay based on estimated arrival rate. Theoretically, the expected queuing delay $C$ is target performance that achieved through periodically optimization in (3) and it should satisfy $C \in \mathbb{R}^+$. However, one major limitation for minimizing local queuing delay is long-term key generation rate. For a consistent traffic flow, continuous key generation process, the minimum queuing delay is the time interval between equivalently continuous key generation event, i.e., $1/\bar{\lambda}_n^m(t)$. Thus, to guarantee a stable queuing system, $C \geq 1/\bar{\lambda}_n^m(t)$ should be satisfied. Constraints (C1) and (C3) come from Lemma 1, let $t = t+T$ and we can have available range for next key supply rate $s_n^m(t+T)$ considering historical status. Constraint (C2) indicates that utilization ratio should be greater than 1 to remain the stability of the queue, and constraint (C4) represents the key supply rate cannot exceed the maximum accepted capacity for burst traffic in an elastic window. Constraint (C5) limits the maximum key supply rate cannot exceed average key consumption rate that exhausts key storage before next key generation event at time $t_n^{m,i+1}$, which can be predicted by $t_n^{m,i+1} = \max \mathbf{t}_n^m$. Constraint (C6) describes the basic constraint in queuing system, i.e., queuing length equals to the queuing length at previous system. Finally, constraint (C7) implies that update interval $T$ must be greater than the expected queuing delay $C$ since $T$ determines the minimum time granularity of key supply rate control on QKD node, otherwise $C$ cannot be converged in practical implementation. Also, constraint (C7) indicates key supply rate should be always greater than 0. In classical networks, the stochastic key requests from users trigger highly dynamic traffic pattern and thus traffic arrival rate becomes unpredictable. Fortunately, the key request traffic can be easily probed in the proposed key supply and rate control scheme, the average traffic arrival rate $a_n^m(\Delta t)$ for $n$-th node can be estimated through periodic probe mechanism, the specific key request traffic arrival rate estimation method is described in Sec. V-A.

The problem formulated in (3) is a Linear Programming (LP) problem, which can be easily resolved by LP solver with polynomial time complexity, e.g., CVX [42] and PuLP [43]. Furthermore, we propose an elAstic key sUpply raTe cOntrol (AUTO) algorithm as described in Alg. 1. By implementing AUTO at each egress port in a decentralized manner, discrete and dynamic key generation process can be concealed behind a periodical port capacity adjustment.

**Algorithm 1:** AUTO

```
1  while System Running do
2      Update length L_n^m(t) of key storage;
3      if Key generation happens or t ≥ t_last + T then
4          # Status Update:
5          if Key generation happens then
6              Update record t_n^m, Λ_n^m(t);
7          end
8          Update history record DoE_n^m(t), ā_n^m(t),
             S̄_n^m(t), Q̄_n^m(t), D̄_n^m(t);
9          t_last = t;
10         # Key Supply Rate Update:
11         s_n^m(Δt) ← Resolve LP problem in (3);
12         # Per-flow Status Update:
13         for f ∈ F_n^m do
14             PauseTh_n^m(f) ← Calculate Eq. (4);
15             if LiveLabel(f) == False &
                 Q̃_n^{m,f}(t) == ∅ then
16                 Release f-th flow queue;
17             end
18             LiveLabel(f) = False;
19         end
20     end
21 end
```

### D. Complexity Analysis

The computational complexity of Alg. 1 can be analyzed as follows. Since each QKD node continuously runs Alg. 1, thus we focus on the iteration from line 3 to line 20. At first step, i.e., status update, the QKD node updates corresponding status and the worst complexity is $\mathcal{O}(1)$. At second step, i.e., key supply rate update, the worst complexity of solving LP problem by using interior-point methods [44] in (3) is $\mathcal{O}(n^{3.5} \cdot L)$, where $n$ equals to the number of variables and $L$ denotes the size of the input data. Since each egress port is isolated, i.e., $n = 1, L = 1$, the worst complexity of solving LP problem for each port is $\mathcal{O}(1)$. At third step, i.e., there are $|\mathbf{F}_n^m|$ maximum iteration times, and the worst complexity in each iteration is $\mathcal{O}(1)$. In total, the computational complexity of Alg. 1 is $\mathcal{O}(|\mathbf{F}_n^m|)$ for $m$-th port on $n$-th QKD node.

## V. Capacity Probing-driven Backpressure Flow Control

In this section, we propose a Capacity Probing-driven Backpressure Flow Control (CP-BFC) scheme to address *unfair resource usage* challenge. At first, we introduce a periodical source probing mechanism to acquire precise port capacity along the routing path. After that, we further propose a backpressure flow control scheme based on the probing information to achieve effective congestion control in QKD networks.

**Algorithm 2:** Periodical Source Probing

FlowStart($f$);
Let probe signal $\mathcal{G}(f) = \emptyset$;
*#Sender Operation:*
**while** $LiveLabel(f) == True$ *or* $Q^{m,f}_{source} \neq 0$ **do**
  **if** *traffic sender receives ACK signal* **then**
    $\mathcal{G}(f) = \mathcal{G}(f) \bigcup \{S^{m,f}_n(t) | n = source\}$;
    SendProbe($\mathcal{G}(f)$) to downstream port;
  **end**
**end**
*#Receiver Operation:*
**while** $LiveLabel(f) == True$ *or* $Q^{m,f}_{source} \neq 0$ **do**
  **if** *traffic receiver receives probe signal* **then**
    SendACK($\mathcal{G}(f)$) to source;
  **end**
**end**
*#Intermediate Operation:*
**while** $LiveLabel(f) == True$ *or* $Q^{m,f}_{source} \neq 0$ **do**
  **if** *node* $n \in \mathcal{P}(f)$ *receives probe signal* $\mathcal{G}(f)$ **then**
    Lookup upstream port capacity from probe signal $\mathcal{G}(f)$ and obtain $S^{m',f}_{upstream}(t)$;
    $\mathcal{G}(f) = \mathcal{G}(f) \bigcup \{S^{m,f}_n(t)\}$;
    SendProbe($\mathcal{G}(f)$) to downstream port;
  **end**
**end**

**Algorithm 3:** Backpressure Flow Control

*# Per-flow Queue Update:*
**if** *Message* $M^f$ *arrives* **then**
  **if** $M^f \notin \mathbf{F}^m_n$ **then**
    Update pause threshold in Eq. (4);
    **if** $L^m_n(t) - \sum_{f \in \mathbf{F}^m_n} \tilde{B}^{m,f}_n \leq PauseTh^m_n(f)$ **then**
      SendPause($f$);
    **end**
    Create a new per-flow queue for $f$-th flow;
    $\tilde{\mathbf{Q}}^{m,f}_n = \tilde{\mathbf{Q}}^{m,f}_n \cup \{M^f\}$;
  **end**
  *# Enqueue Event:*
  **else**
    $\tilde{\mathbf{Q}}^{m,f}_n = \tilde{\mathbf{Q}}^{m,f}_n \cup \{M^f\}$;
    **if** $|\tilde{\mathbf{Q}}^{m,f}_n| \geq \tilde{B}^{m,f}_n - PauseTh^m_n(f)$ **then**
      $PausLabel(f) = True$;
      $LiveLabel(f) = Ture$;
      SendPause($f$);
    **end**
  **end**
**end**
*# Dequeue Event:*
**if** *Message* $M^f$ *departs from* $\tilde{\mathbf{Q}}^{m,f}_n$ **then**
  $\tilde{\mathbf{Q}}^{m,f}_n = \tilde{\mathbf{Q}}^{m,f}_n - \{M^f\}$;
  **if** $PausLabel(f) == True$ & $|\tilde{\mathbf{Q}}^{m,f}_n| < \tilde{B}^{m,f}_n - PauseTh^m_n(f)$ **then**
    SendResume($f$);
    $PausLabel(f) = False$;
  **end**
**end**

### A. Periodical Source Probing

AUTO algorithm periodically adjusts key supply rate, i.e., egress port capacity, on each QKD node, and thus egress port capacity remains stable within a time duration between adjacent key generation event, which is a relatively large time scale compared to round-trip time for classical signals. To accurately estimate the capacity and avoid radical traffic entry from upstream port, we further design a Periodical Source Probing (PSP) mechanism for per-flow control. PSP can assist relay nodes to update upstream port capacity information. The specific PSP mechanism is described in Alg. 2. Source node of each flow periodically sends probe signal to collect real-time port capacity after flow starts. For each relay node along the routing path $\mathcal{P}(f) = \{n | n \in \mathcal{N}\}$, it updates upstream port capacity for pause threshold calculation (this concept will be introduced in the next subsection), and calculates the bottleneck port capacity, i.e., key supply rate at corresponding egress port, and sets traffic arrival rate as $a^m_n(t+T) = \sum_{f \in \mathbf{F}^m_n} \min\{s^{m'}_{n'}(t) | n' \in \mathcal{P}(f), m' \in \mathcal{M}(f)\}$, where $\mathbf{F}^m_n$ represents the flow set for $m$-th port on $n$-th node and $\mathcal{M}(f)$ represents the egress port set along the routing path for $f$-th flow. As long as a flow remains alive, PSP continues on source node as long as a flow remains alive or per-flow queue is non-empty.

### B. Backpressure Flow Control

To avoid radical traffic entry and achieve a fair resource usage among multiple users, an effective congestion control is required. Inspired by previous packet loss-free design in classical networks, e.g., ATM and modern data center per-hop per-flow flow control system [18, 45, 46], we further propose a backpressure flow control scheme in QKD network. As shown in the proposed KM&S framework, per-flow queues are maintained by each QKD node, which should be processed after routing module. Meanwhile, a pause threshold $PauseTh$ is set for each per-flow queue, which is introduced to avoid message loss due to queue overflow. Once the key request messages in the per-flow queue exceeds $PauseTh$, a pause signal should be sent to upstream port to temporarily stop message forwarding.

The basic idea of the backpressure flow control can be described as follows. At first, a QKD node maintains per-flow queues and state at each egress port. While a new key request flow arrives, referred KM&S framework as shown in Fig. 5, the corresponding key request messages are forwarded to a egress port after authentication and route decision, and a new per-flow queue is created at the port. Second, the service state of per-flow queues on each egress port is independently controlled based on its real-time status. For each per-flow queue, it forwards key request message in FIFO order. While every incoming message belongs to flow $f$ arrives (enqueue event), the status of per-flow queue that belongs to $f$ should be checked. Once the queue length exceeds the pause threshold,

a pause signal should be sent to upstream port immediately. On the contrary, once the queue length down below the pause threshold after a message departure (dequeue event), a resume signal should be sent. In this case, each flow can be precisely controlled per-hop.

The proposed backpressure flow control pauses flow $f$ if the occupancy of the queue assigned to flow $f$ exceeds the pause threshold $PauseTh$, which is set as 1-hop BDP at the queue drain rate as follow.

$$PauseTh_n^m(f) = RTT^{1-hop} \cdot \frac{S_{n,upstream}^{m',f}(t)}{F_{active}} \cdot \frac{w_{n'}^{m',f}}{W_{n'}^{m'}}, \quad (4)$$

where $RTT^{1-hop}$ denotes 1-hop Round-Trip Time (RTT) to the upstream port, $F_{active}$ denotes the number of active per-flow queues at an egress port, i.e., non-empty queues with messages to transmit that are not paused, $S_{n,upstream}^{m',f}(t)$ denotes upstream port capacity at time $t$, and $w_{n'}^{m',f}$ denotes scheduling weight for flow $f$. Eq. (4) considers a weighted scheduling, and $W_{n'}^{m'} = \sum_{f \in \mathbf{F}_{n'}^{m'}} w_{n'}^{m',f}$. If a fair scheduling such as round-robin is adopted, then $w_{n'}^{m',f} = 1$ for each flow.

The specific backpressure flow control is described in Alg. 3, which are triggered by two events, i.e., message forwarding event at egress port and key generation event at key storage. For message forwarding event, while a new message $M^f$ arrives, the egress port checks its related flow information according to flow ID[2]. If $M^f$ belongs to a flow that has been locally recorded, the egress port checks if the corresponding queue length exceeds pause threshold after enqueue event as described in lines 11-22. Otherwise, the egress port creates a new per-flow queue for $f$-th flow and calculate the corresponding pause threshold in lines 4-10. It should be noted that pause signal is sent back to upstream port once the buffer size exhausts and the remaining buffer size is short to support a new queue creation with the size of pause threshold. For key generation event, key supply rate should be updated as described in Alg. 1. If an update of key supply rate happens at upstream port, an update of pause threshold should also be triggered as described in Eq. (4) at downstream port once next probe signal arrives. Additionally, all per-flow queues at egress port are checked and corresponding queues of invalid flows are discarded in lines 13-19 in Alg. 1.

To avoid unnecessary message backlog, as described in Sec IV, per-flow queue cannot exceed the real-time key storage volume $L_n^m(t)$ at time $t$. Thus, constraint $\sum_{f \in \mathbf{F}_n^m} |\tilde{\mathbf{Q}}_n^{m,f}| \leq L_n^m(t)$ should be satisfied. Let $\tilde{B}_n^{m,f}$ denote the available length limit of per-flow queue for $f$-th flow at $m$-th port on $n$-th node. To determine $\tilde{B}_n^{m,f}$, a weighted allocation method is adopted and thus $\tilde{B}_n^{m,f} = \frac{L_n^m(t)}{|\mathbf{F}_n^m|+1}$, which is actually a dynamic length limit and $\tilde{B}_n^{m,f} \geq |\tilde{\mathbf{Q}}_n^{m,f}|$ should only be guaranteed at the moment of per-flow queue creation. Once a per-flow queue length $|\tilde{\mathbf{Q}}_n^{m,f}| \geq \tilde{B}_n^{m,f} - PauseTh_n^m(f)$ is satisfied in lines 14-18, a pause signal is sent to upstream port.

[2]Each flow has a unique flow identifier, which is indicated by a 4-tuple of the source and destination addresses, port number in TCP/IP stack, and protocol. The hash value of the flow ID is adopted to track identify the flow in memory space [18].

### C. Complexity Analysis

In Alg. 2, the worst complexity of lookup operation in line 19 is $\mathcal{O}(\max_f |\mathcal{P}(f)|)$, which is determined by the maximum length of routing path. The other operations is $\mathcal{O}(1)$, the computational complexity of Alg. 2 can be calculated as $\mathcal{O}(\max_f |\mathcal{P}(f)|)$ for each port on the QKD node. The computational complexity of Alg. 3 can be analyzed as follows. At first step, i.e., *per-flow queue update*, the complexity of lookup operation for a list in line 3 is $\mathcal{O}(|\mathbf{F}_n^m|)$, the creation of per-flow queue related to the memory allocation and the complexity is $\mathcal{O}(1)$. Thus, the worst complexity from line 3 to line 10 is $\mathcal{O}(1)$. At second step, i.e., *enqueue event*, the worst complexity from line 12 to line 19 is determined by the enqueue operation, i.e., $\mathcal{O}(\max_f |\tilde{\mathbf{Q}}_n^{m,f}|)$, where $\max_f |\tilde{\mathbf{Q}}_n^{m,f}|$ represents the maximum message number in a per-flow queue. At third step, i.e., *dequeue event*, the worst complexity from line 23 to line 27 is $\mathcal{O}(1)$. In total, the computational complexity of Alg. 3 is $\mathcal{O}(|\mathbf{F}_n^m| + \max_f |\tilde{\mathbf{Q}}_n^{m,f}|)$ for $m$-th port on $n$-th QKD node, which is determined by the total number of alive traffic flows and undeparted messages in per-flow queue.

## VI. Experimental Implementation

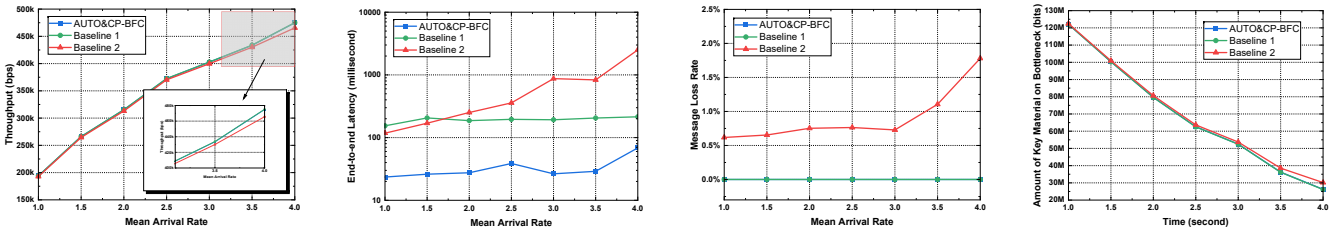
In this section, we implement the proposed KM&S framework in an overlay network and conduct extensive experiments compared to representative schemes. The specific experimental settings and results are introduced, respectively.

TABLE II
Experimental Parameters

| Parameter | Value |
|---|---|
| Network scale | 12 |
| Optical link attenuation | 10km@3.33dB<br>20km@5.35dB |
| The long-term key generation rate (scaling factor=10) | 50kbps×10 |
| Classical channel capacity | 1Gbps |
| Unit length of key material & message size | 512 bytes |
| Update interval $T$ | 8ms |
| Expected queuing delay $C$ | 8ms |
| Elastic window size $W$ | 5 |
| Convergence gap $\epsilon$ | 0 |
| Key request scheduler | Round-robin |
| One-hop link propagation delay | 0.1 ms |
| Experiment duration | 10 min |

### A. Experimental Setting

To prove the feasibility of KM&S framework and the superiority of the proposed schemes, we conduct extensive experiments through overlaying implementation on classical switches attached to a pair of realistic QKD devices, and the specific QKD device model is QKD-PHA1250 produced by QuantumCTek company [27]. Since the realistic QKD device output a batch of packets to key management terminal through private interface and the unit secret-key contained in each packet is 512 bytes, the unit secret-key is also set as $Length = 512bytes$ for length alignment. It should be noted that arbitrary $Length$ is allowed in practical implementation. To present comprehensive performance comparisons,




(a) Throughput (b) Latency (c) Message loss (d) Key storage

Fig. 7. Performance comparison versus mean arrival rate in PPBP model under single user scenario (mean arrival rate in PPBP = $[1, 4]$).

representative schemes in existing studies are adopted in our experiments, we introduce them as follows.

- **Baseline 1:** QKD-Transport Layer (TL) protocol in SECOQC [8] realizes an end-to-end transport between source node and destination node in QKD networks. To prevent network congestion, QKD-TL pro-actively react to key storage shortage event on relay nodes. We define a key storage shortage event happens when $DoE_n^m(t) \leq 1$ at $m$-th port on $n$-th node, and notification signal from $n$-th node to source node is realized by using ECN mechanism [47, 48], which reduces the congestion window size at source node to prevent the exhaustion of the key storage on the relay node. Specifically, the slow start threshold in QKD-TL is set as the same as the long-term key generation rate, i.e., 500 kbps, and ECN threshold is set as 800kbits.
- **Baseline 2:** By sending notification signals to congestion control mechanism, Active Queue Management (AQM) can manage the size of queues and avoid congestion or queuing delay in advance [49]. To manage the precious resource in key storage, we enable a classical AQM algorithm to adjust key supply rate, i.e., random early detection [50, 51], and adopt TCP Reno as congestion control algorithm for end-to-end key exchange process. The specific key supply rate adjustment follows

$$s_n^m(t) = \begin{cases} s^{min}, & L_n^m(t) < \tau^{min}, \\ f(s^{min}, L_n^m(t)), & \tau^{max} \leq L_n^m(t) < \tau^{min}, \\ s^{max}, & \tau^{max} \leq L_n^m(t), \end{cases} \tag{5}$$

where $f(s^{min}, L_n^m(t)) = s^{min} + \frac{s^{max}(L_n^m(t) - \tau^{min})}{k(\tau^{max} - \tau^{min})}$ is a non-increasing function of $L_n^m(t)$, $s^{min} = s^{max}/k$ and we take typical value of $k = 2$, and $\tau^{min} = 800$ kbits and $\tau^{max} = 16$ Mbits are upper threshold and lower threshold, respectively.

To verify the effectiveness of key management and congestion control and eliminate the impact of dynamic routing, Open Shortest Path First (OSPF) routing protocol and round-robin scheduler are implemented by default. To guarantee the reliability of the transmission, all key request messages and tailored signals (including probe, pause, and resume signal) between adjacent QKD nodes are transmitted through TCP stream socket since an overlaying implementation is conducted. Meanwhile, since two baselines both adopt TCP-like congestion control mechanism, we set a maximum rate limitation as 1Mbps, which is two-fold long-term key generation rate, for all users to avoid rapid increasing of congestion window and the exhaustion of limited key material in key storage. It should be noted that $\lambda_n^m(t)$ refers to real-time key generation rate as we described in theoretical analysis. Since we conduct experiments equipped with realistic QKD devices, $\lambda_n^m(t)$ is determined by the QKD device in real-time instead of pre-defined parameter. Due to the limited key generation rate in realistic from open-source dataset (approximately 50kbps over 10km optical fiber) [26], the available unit secret-keys become scarce and the reliability of experimental results might not be supported. Thus, we conduct a scaling technique and choose 10 as the scaling factor. As a result, the long-term key generation rate is increased from 50kbps to 500kbps. To ensure network service stability, the setting of expected queuing delay $C$ should consider the key generation rate. Since the long-term key generation rate between adjacent QKD nodes is 500 kbps, which indicates that an unit secret-key generation interval is 8 ms on average for a QKD node. To maintain the consistency between the time taken to generate an unit secret-key and supply one to serve key request, we align the expected queuing delay $C$ with the secret-key generation interval, i.e., $C = 8ms$. As described in the Sec. IV, the update interval $T$ and expected queuing delay $C$ should satisfy constraint $T \geq C$. For simplicity, we let $T = C$ in our experiments. For the elastic window information updates, they are triggered by key generation events. After each key generation event (i.e., a batch of secret-keys arrive), the amount of key material information in key storage is updated. Consequently, the elastic window size $W$ is set to 5 arrivals of key generation event. Other experimental parameter settings are listed in Table II.

In our experiment, we adopt a China metropolitan-area network topology [52, Fig. 1] as network topology, which consists of three subnetworks that are directly connected to each other through three backbone trusted relays. Since we only consider trusted relay-based interconnection approach, 3 users connected through a trusted relay are actually deployed in each subnetwork, and there are 12 nodes in total as shown in Fig. 8. Open-source codes and instruction documents are provided in [53].

## B. Results

*1) Single User Scenario:* To evaluate the behavior in terms of consistency, we construct single user scenario and conduct performance comparisons between the proposed scheme and

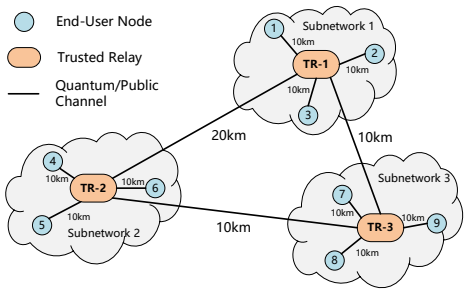
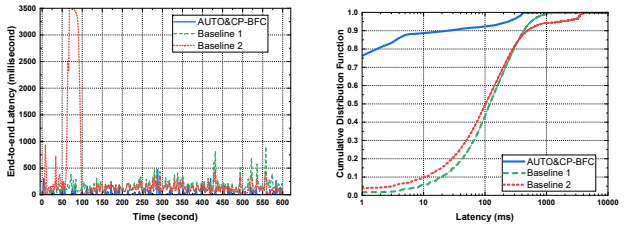



Fig. 8. Illustration of network topology deployed in the experiment.



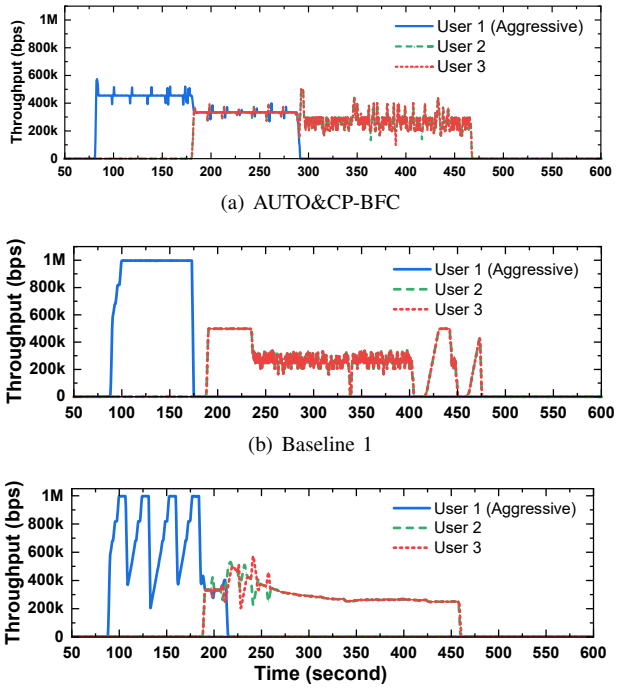

(a) Temporal distribution (b) Cumulative distribution function

Fig. 9. Temporal distribution and cumulative distribution in terms of end-to-end latency under single user scenario (mean arrival rate in PPBP = 1.5).

baselines. We adopt Poisson Pareto Burst Process (PPBP)[3] [16] as a traffic source generator to inject key request traffic, and parameter setting follows: the traffic type is bursty, Hurst parameter is 0.8, and Pareto shape parameter (ON-period) is 1.2. Fig. 7 illustrates the statistical results in terms of throughput, end-to-end latency, and message loss rate. The traffic intensity is adjusted through the variation of mean arrival rate in PPBP-based traffic generator, and there are 5 seconds as initial duration for the QKD network before user starts key requests. As we can see, the proposed AUTO&CP-BFC scheme shows significant superiority in terms of end-to-end latency while achieves similar performance in terms of throughput. For message packet loss, the proposed scheme can effectively avoid congestion loss since CP-BFC scheme never inject excessive amount of traffic beyond the length $L_n^m(t)$ of key storage. Baseline 1 can also avoid congestion loss by proactively reacting to key storage shortage via ECN mechanism. However, since congestion window continuous increases if no ECN is received, congestion loss is still possible for baseline 1 especially when sufficient key material are stored in advance and switches are equipped with small buffer. Baseline 2, which decouples local key management from AQM and congestion control from TCP, frequently triggers congestion loss due to the mismatch between port capacity decreasing process and congestion window increasing process.

Fig. 9 illustrates the detailed results in terms of end-to-end latency. Since AUTO determines key supply rate through optimizing queuing delay, which is the dominated part in end-to-end latency, the proposed schemes achieve consistent behavior throughout the experiment especially for tail latency.

[3]PPBP model could be considered a limiting case for the multiplexing of a large number of such independent heavy-tailed on-off sources [54]. Thus, PPBP appears a natural candidate for the modeling of bursty packet data traffic streams commonly observed in real-world network traffic.




(a) AUTO&CP-BFC

(b) Baseline 1

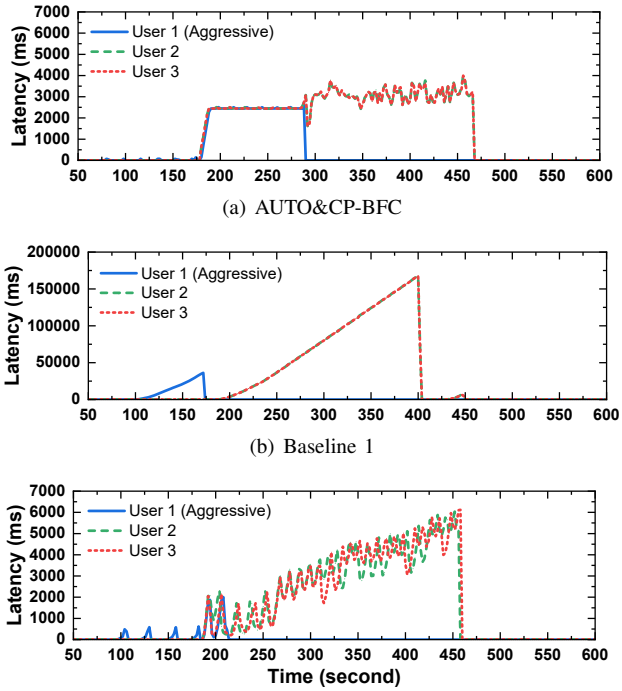
(c) Baseline 2

Fig. 10. Performance comparisons in terms of throughput (traffic size per flow = 80Mb, duration of early traffic entry = 100s).

As a comparison, baseline 1 periodically experiences latency increasing during the experiment, this phenomenon stems from greedy traffic entry behavior of TCP and thus excessive key request messages are accumulated in the queue on relay nodes until an ECN is received. Even if the key material are sufficient for traffic, messages injected into the network still have to wait a relative long queuing delay resulting in longer latency. Meanwhile, baseline 2 experiences a significant latency increasing in the first two minutes. The reason can be explained as follows. For baseline 2, AQM can manage key material in key storage and effectively avoid resource exhaustion, but the sending rate on source node keep increasing until the queue length approaches buffer size and incurs ECN. In this context, baseline 2 can easily experience rapid latency increases especially when the resource shortage in key storage happens or buffer size is adopted on switches.

*2) Multiple User Scenario:* In the next, we construct a typical multiple user scenario to present fairness issues facing "first-arrival advantage" and excessive traffic entry, respectively. We conduct performance comparisons to evaluate the behavior in terms of both consistency and fairness, and adopt an ON-OFF traffic generator to continuously inject traffic at users for a period. In each subnetwork, one aggressive user is deployed and other users follow the same traffic pattern.

The typical scenario is that aggressive user (i.e., user 1) always injects key request traffic before other users. Fig. 10 and Fig. 11 illustrate the performance comparisons in terms of throughput and end-to-end latency, respectively. Since the proposed scheme adopts AUTO to periodically adjust key supply rate and stabilize queuing delay, early traffic entry from aggressive users cannot acquire more resources, redundant key

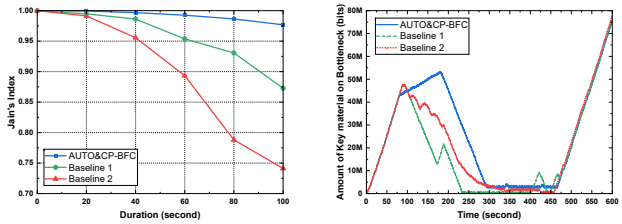



(a) AUTO&CP-BFC

(b) Baseline 1

(c) Baseline 2

Fig. 11. Performance comparisons in terms of end-to-end latency (traffic size per flow = 80Mb, duration of early traffic entry = 100s).



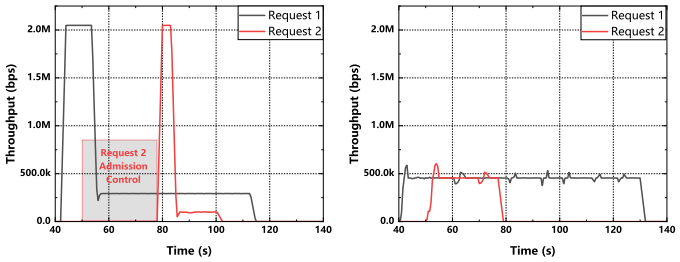

(a) Fairness

(b) Key storage status

Fig. 12. Fairness comparison versus the duration of early traffic entry and real-time key storage status on bottleneck node (traffic size per flow = 80Mb).

material in key storage is not available until multiple users (i.e., user 2 and user 3) arrive, and thus it can effectively prevent "first-arrival advantage" and guarantee fair resource usage among multiple users in each subnetwork. As a comparison, results for two baselines show a typical "first-arrival advantage" for the aggressive user. Inherently, baseline 1 and baseline 2 adopt on-demand key supply strategy, even through they also adopt fair scheduling strategy (i.e., round-robin) for multiple users, the behavior of early traffic entry still acquires higher performance in terms of throughput and latency than other users. Note that other typical scenarios, e.g., aggressive user always injects more traffic than other users, can be easily constructed and similar results are expected.

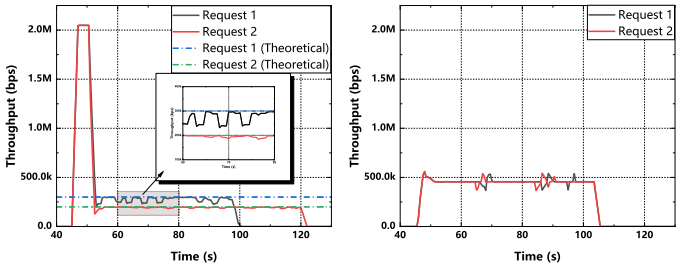
Fig. 12(a) illustrates fairness results by using Jain's index [55]. As we can see, Jain's index of baseline 1 and baseline 2 decreases while the duration of early traffic entry behavior increases, yet the proposed scheme consistently achieves near-optimal fairness regardless of the duration of early traffic entry behavior. Fig. 12(b) illustrates the real-time key storage status




(a) RSVP

(b) The Proposed Scheme

Fig. 13. Negative impact of admission control.




(a) RSVP

(b) The Proposed Scheme

Fig. 14. Performance degradation in practical implementation.

of the proposed scheme and two baselines on bottleneck node, and the proposed scheme can maintain DoE above 1 even when there are multiple users' competition traffic.

*3) Extended Scenario:* Since similar designs of per-flow-based protocols are also considered in existing studies, an extended comparison between AUTO&CP-BFC scheme and the representative scheme RSVP is further conducted. The parameters and request model definitions follows [21, Chapter], and the service rate $R$ can be calculated as Eq. (3.12) in [21]. The relevant parameter settings are shown in Table III. To illustrate this issue, we choose user 4→user 7 (flow 1) and user 5→user 8 (flow 2) to generate two traffic flows.

In short, RSVP exists two shortcomings: 1. Negative impact of admission control. RVSP requires admission control for requests, which reduces the flexibility of service. For RSVP, as shown in Fig. 13, the admission of flow 1 reserves a significant portion of key resources from the key storage. This resource reservation leads to insufficient resources for subsequent key request traffic, and thus flow 2 experiences long admission delays. Additionally, RSVP initially leverages reversed resources to serve requests, ensuring high throughput for a certain period. However, once the reversed resources are exhausted, the throughput of ongoing key requests drops dramatically, reverting to the reversed bucket rate $r$. This phenomenon is referred to as the cliff effect in this paper. In contrast, the proposed scheme dynamically adjusts the key supply rate of requests, avoiding resource exhaustion caused by servicing a single request. This ensures that flow 2 is serviced promptly without waiting for significant resource replenishment without reserving key storage resources. 2. Performance degradation in practical implementation. The variability in the key generation rate of the links can lead to the failure of resource reservation,

TABLE III
PARAMETER SETTINGS FOR FIG. 13(LEFT) AND FIG. 14(RIGHT)

| | Flow 1/2 | Flow 1/2 |
|---|---|---|
| Bucket rate $r$ | 300/100kbps | 300/200kbps |
| Peak rate $p$ | 4/4Mbps | 4/4Mbps |
| Bucket depth $b$ | 10/5Mb | 5/5Mb |
| Max packet size $M$ | 4/4kb | 4/4kb |
| Start time | 40/50s | 45/45s |

rendering it incapable of providing the Quality of Service (QoS) guarantees for active requests. To illustrate this issue, we selected two concurrent requests for service. For RSVP, the average key generation rate of the link is 500 kbps, and during the admission of flow 1 and flow 2, it can satisfy the resource reservation requirements. However, due to the variability of the key generation rate as illustrated in Fig. 1, when the rate drops below 500 kbps, the throughput for the requests fluctuates, as shown in Figure 14(a), even with resource reservation. In contrast, the proposed scheme takes the fluctuation in the key generation rate into account. By dynamically adjusting the key supply rate and other parameters, it helps maintain the stability of throughput, as shown in Figure 14(b). Therefore, relying solely on resource reservation, as done in the RSVP scheme, cannot effectively solve the issue of performance degradation in practical implementation.

## VII. CONCLUSION

In this paper, we concentrated on decentralized multi-hop QKD networks, and pointed out critical challenges triggered by unique characteristics of quantum key material. Motivated by the proof of preliminary experiments, at first, we devised a KM&S framework to support pipeline processing and decoupled function through both overlaying and underlying implementation in TCP/IP protocol stack. Second, the periodically local key supply rate adjustment is considered to address the fairness challenge among multiple users but also elasticity for burst traffic, and we proposed an elastic key supply rate control scheme, named AUTO, to provide both elasticity and fairness. Third, end-to-end congestion control is considered to address inaccurate estimation and performance consistency challenges, and we proposed CP-BFC scheme to avoid congestion loss and cliff effect. We implemented KM&S framework in overlaying IP networks assisted by realistic QKD devices, and conducted extensive experiments to prove the superiority of the proposed schemes compared to representative schemes in existing studies. Results demonstrated that the proposed can provide consistent performance especially end-to-end latency and guarantee fairness among multiple users.

Although numerous practical problems still remain, our work is a pioneering attempt to improve the quality of service in decentralized QKD networks and pave the way for the QKD network implementation. To facilitate readers to follow our work, we also provide open-source code and dataset for open collaboration. The specific functional design in KM&S framework such as key request scheduler and resource allocation is worthy of further studying. In the future development, an evolving architecture integrating QKD networks and IP networks is expected to provide security service with QoS requirement, we believe this work can provide theoretical guidance for future deployment of decentralized multi-hop QKD networks.

## ACKNOWLEDGMENT

This work is supported in part by the National Natural Science Foundation of China under Grant 62201540 and 6240246, in part by the Innovation Program for Quantum Science and Technology under Grant No.2021ZD0301301, in part by the Anhui Initiative in Quantum Information Technologies under grant No. AHY150400, and in part by the Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant No. Y202093.

## APPENDIX A
## PROOF OF LEMMA 1

*Proof.* The convergence constraint in Ineq. (2) can be further rewritten as:

$$-\varepsilon \leq \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \frac{Q(t-T)-S(t)}{a(t)} + T - C \leq \varepsilon. \quad (6)$$

We consider the upper bound in Ineq. (6) at first, then the constraint can be rewritten as:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \frac{Q(t-T)-S(t)}{a(t)} \leq \varepsilon - (T-C),$$

$$\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} S(t) \geq \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} Q(t-T) - (T-C-\varepsilon) \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} a(t).$$

If we consider long-term stability, all status of the queue can be taken as average value and thus we have $\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} S(t) = \bar{S}(t)$, $\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} Q(t-T) = \bar{Q}(t-T)$, and $\lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} a(t) = \bar{a}(t)$. Meanwhile, the constraint can be further rewritten as:

$$\bar{S}(t) \geq \bar{Q}(t-T) - (T-C-\varepsilon)\bar{a}(t).$$

Similarly, considering the lower bound in Ineq. (6), we can have:

$$\bar{S}(t) \leq \bar{Q}(t-T) + (T-C+\varepsilon)\bar{a}(t).$$

Thus, the proof is complete. □

## REFERENCES

[1] Z. Li, K. Xue, J. Li, L. Chen, R. Li, Z. Wang, N. Yu, D. S. Wei, Q. Sun, and J. Lu, "Entanglement-assisted quantum networks: Mechanics, enabling technologies, challenges, and research directions," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2133–2189, 2023.

[2] L.-J. Wang, K.-Y. Zhang, J.-Y. Wang, J. Cheng, Y.-H. Yang, S.-B. Tang, D. Yan, Y.-L. Tang, Z. Liu, Y. Yu *et al.*, "Experimental authentication of quantum key distribution with post-quantum cryptography," *npj Quantum Information*, vol. 7, no. 1, pp. 1–7, 2021.

[3] J. Li, M. Wang, K. Xue, R. Li, N. Yu, Q. Sun, and J. Lu, "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Transactions on Communications*, 2022.

[4] H.-K. Lo, X. Ma, and K. Chen, "Decoy state quantum key distribution," *Physical Review Letters*, vol. 94, no. 23, p. 230504, 2005.

[5] X. Ma, B. Qi, Y. Zhao, and H.-K. Lo, "Practical decoy state for quantum key distribution," *Physical Review A*, vol. 72, no. 1, p. 012326, 2005.

[6] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, "Experimental quantum cryptography," *Journal of cryptology*, vol. 5, no. 1, pp. 3–28, 1992.

[7] C. Elliott, A. Colvin, D. Pearson, O. Pikalo, J. Schlafer, and H. Yeh, "Current status of the DARPA quantum network," in *Quantum Information and computation III*, vol. 5815. SPIE, 2005, pp. 138–149.

[8] M. Peev, C. Pacher, R. Alléaume, C. Barreiro, J. Bouda, W. Boxleitner, T. Debuisschert, E. Diamanti, M. Dianati, J. Dynes *et al.*, "The SECOQC quantum key distribution network in Vienna," *New Journal of Physics*, vol. 11, no. 7, p. 075001, 2009.

[9] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka *et al.*, "Field test of quantum key distribution in the Tokyo QKD network," *Optics Express*, vol. 19, no. 11, pp. 10 387–10 409, 2011.

[10] Y.-A. Chen, Q. Zhang, T.-Y. Chen, W.-Q. Cai, S.-K. Liao, J. Zhang, K. Chen, J. Yin, J.-G. Ren, Z. Chen *et al.*, "An integrated space-to-ground quantum communication network over 4,600 kilometres," *Nature*, vol. 589, no. 7841, pp. 214–219, 2021.

[11] M. Mehic, M. Niemiec, S. Rass, J. Ma, M. Peev, A. Aguado, V. Martin, S. Schauer, A. Poppe, C. Pacher *et al.*, "Quantum key distribution: a networking perspective," *ACM Computing Surveys (CSUR)*, vol. 53, no. 5, pp. 1–41, 2020.

[12] J. Schmid, A. Höss, and B. W. Schuller, "A survey on client throughput prediction algorithms in wired and wireless networks," *ACM Computing Surveys (CSUR)*, vol. 54, no. 9, pp. 1–33, 2021.

[13] G. Abbas, Z. Halim, and Z. H. Abbas, "Fairness-driven queue management: A survey and taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 324–367, 2015.

[14] 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 Transactions on Networking*, vol. 28, no. 1, pp. 168–181, 2019.

[15] H. Zhou, K. Lv, L. Huang, and X. Ma, "Quantum network: security assessment and key management," *IEEE/ACM Transactions on Networking*, vol. 30, no. 3, pp. 1328–1339, 2022.

[16] M. S. Akhtar, G. Krishnakumar, B. Vishnu, and A. Sinha, "Fast and secure routing algorithms for quantum key distribution networks," *IEEE/ACM Transactions on Networking*, vol. 31, no. 5, pp. 2281–2296, 2023.

[17] R. Al-Saadi, G. Armitage, J. But, and P. Branch, "A survey of delay-based and hybrid TCP congestion control algorithms," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3609–3638, 2019.

[18] P. Goyal, P. Shah, K. Zhao, G. Nikolaidis, M. Alizadeh, and T. E. Anderson, "Backpressure flow control," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, 2022, pp. 779–805.

[19] S. Wang, W. Chen, Z.-Q. Yin, H.-W. Li, D.-Y. He, Y.-H. Li, Z. Zhou, X.-T. Song, F.-Y. Li, D. Wang *et al.*, "Field and long-term demonstration of a wide area quantum key distribution network," *Optics Express*, vol. 22, no. 18, pp. 21 739–21 756, 2014.

[20] S.-K. Liao, W.-Q. Cai, J. Handsteiner, B. Liu, J. Yin, L. Zhang, D. Rauch, M. Fink, J.-G. Ren, W.-Y. Liu *et al.*, "Satellite-relayed intercontinental quantum network," *Physical Review Letters*, vol. 120, no. 3, p. 030501, 2018.

[21] M. Mehic, S. Rass, P. Fazio, M. Voznak *et al.*, *Quantum Key Distribution Networks: A Quality of Service Perspective*. Springer, 2022.

[22] C. Yang, H. Zhang, and J. Su, "Quantum key distribution network: Optimal secret-key-aware routing method for trust relaying," *China Communications*, vol. 15, no. 2, pp. 33–45, 2018.

[23] S. B. S. H. R. Braden, L. Zhang and S. Jamin, "Resource ReSerVation protocol (RSVP) – version 1 functional specification," Internet Engineering Task Force (IETF), RFC 2205. https://datatracker.ietf.org/doc/html/rfc5340, Tech. Rep., 2008.

[24] M. Mehic, A. Maric, and M. Voznak, "QSIP: A quantum key distribution signaling protocol," in *Multimedia Communications, Services and Security: 9th International Conference, MCSS 2017, Kraków, Poland, November 16-17, 2017, Proceedings 9*. Springer, 2017, pp. 136–147.

[25] Y. Tanizawa, R. Takahashi, H. Sato, A. R. Dixon, and S. Kawamura, "A secure communication network infrastructure based on quantum key distribution technology," *Ieice Transactions on Communications*, vol. 99, no. 5, pp. 1054–1069, 2016.

[26] J. Li, Y. Chen, P. Zheng, Z. Li, L. Chen, and Y. Yang, "Open-source key generation dataset for QKD devices," https://github.com/QNLab-USTC/USTC-KMS24-QKD-Dataset, accessed on Dec., 2024.

[27] L. QuantumCTek Co., "Miniaturized time-bin phase-coding quantum key distribution terminal," https://www.quantum-info.com/English/product/National/2017/0901/324.html, accessed on Dec., 2024.

[28] M. Allman, V. Paxson, ICSI, and E. Blanton, "Tcp congestion control," Internet Engineering Task Force (IETF), RFC 5681. https://datatracker.ietf.org/doc/html/rfc5681, Tech. Rep., 2009.

[29] W. Kozlowski, S. Wehner, R. Van Meter, B. Rijsman, A. Cacciapuoti, M. Caleffi, and S. Nagayama, "Architectural principles for a quantum internet," Internet Engineering Task Force (IETF), RFC 9340. https://datatracker.ietf.org/doc/rfc9340/, Tech. Rep., 2023.

[30] O. Maurhart, C. Pacher, A. Happe, T. Lor, C. Tamas, A. Poppe, and M. Peev, "New release of an open source QKD software: design and implementation of new algorithms, modularization and integration with IPSec," *3rd Annual Conference on Quantum Cryptography*, p. 1, 2013.

[31] J. Li, H. Lu, K. Xue, and Y. Zhang, "Temporal netgrid model-based dynamic routing in large-scale small satellite networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 6009–6021, 2019.

[32] R. Coltun, D. Ferguson, J. Moy, and A. Lindem, "OSPF for IPv6," Internet Engineering Task Force (IETF), RFC 5340. https://datatracker.ietf.org/doc/html/rfc5340, Tech. Rep., 2008.

[33] J. Wu, L. Chen, J. Zhang, Z. Huang, Z. Li, J. Li, K. Xue, and N. Yu, "A distributed routing protocol based on key reservation in quantum key distribution networks," in *2024 IEEE International Conference on Communications (ICC)*, 2024, pp. 1–6.

[34] M. Wang, J. Li, K. Xue, R. Li, N. Yu, Y. Li, Y. Liu, Q. Sun, and J. Lu, "A segment-based multipath distribution method in partially-trusted relay quantum networks," *IEEE Communications Magazine*, vol. 61, no. 12, pp. 184–190, 2023.

[35] H. Wen, Z. Han, Y. Zhao, G. Guo, and P. Hong, "Multiple stochastic paths scheme on partially-trusted relay quantum key distribution network," *Science in China Series F: Information Sciences*, vol. 52, no. 1, pp. 18–22, 2009.

[36] H. M. Chaskar and U. Madhow, "Fair scheduling with tunable latency: a round-robin approach," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 592–601, 2003.

[37] Y. Cao, Y. Zhao, Q. Wang, J. Zhang, S. X. Ng, and L. Hanzo, "The evolution of quantum key distribution networks: On the road to the qinternet," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 839–894, 2022.

[38] L. Chen, K. Xue, J. Li, Z. Li, and N. Yu, "Q-CSKDF: A continuous and security key derivation function for quantum key distribution," *IEEE Network*, vol. 38, no. 5, pp. 123–130, 2024.

[39] G. Li, H. Luo, J. Yu, A. Hu, and J. Wang, "Information-theoretic secure key sharing for wide-area mobile applications," *IEEE Wireless Communications*, vol. 31, no. 1, pp. 118–124, 2023.

[40] J. Yang and S. Ulukus, "Trading rate for balanced queue lengths for network delay minimization," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 988–996, 2011.

[41] V. G. Kulkarni, *Modeling and analysis of stochastic systems*. Chapman and Hall/CRC, 2016.

[42] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," 2014.

[43] A. M. M. O. A. P. Stuart Mitchell, Anita Kean and F. Peschiera, "PuLP: an linear and mixed integer programming modeler," https://coin-or.github.io/pulp/, accessed on Dec., 2024.

[44] Y. Ye, *Interior point algorithms: theory and analysis*. John Wiley & Sons, 2011.

[45] N. Kung and R. Morris, "Credit-based flow control for atm networks," *IEEE Network*, vol. 9, no. 2, pp. 40–48, 1995.

[46] R. Laufer, T. Salonidis, H. Lundgren, and P. Le Guyadec, "A cross-layer backpressure architecture for wireless multihop networks," *IEEE/ACM Transactions on Networking*, vol. 22, no. 2, pp. 363–376, 2013.

[47] K. De Schepper and B. Briscoe, "The explicit congestion notification (ECN) protocol for low latency, low loss, and scalable throughput (l4s)," Internet Engineering Task Force (IETF), RFC 9331. https://datatracker.ietf.org/doc/rfc9331/, Tech. Rep., 2023.

[48] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 689–702, 2003.

[49] E. F. Baker and E. G. Fairhurst, "IETF recommendations regarding active queue management," Internet Engineering Task Force (IETF), RFC 7567. https://datatracker.ietf.org/doc/rfc9331/, Tech. Rep., 2015.

[50] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[51] D. Niyato and E. Hossain, "Queue-aware uplink bandwidth allocation and rate control for polling service in IEEE 802.16 broadband wireless networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 668–679, 2006.

[52] T.-Y. Chen, X. Jiang, S.-B. Tang, L. Zhou, X. Yuan, H. Zhou, J. Wang, Y. Liu, L.-K. Chen, W.-Y. Liu *et al.*, "Implementation of a 46-node quantum metropolitan area network," *npj Quantum Information*, vol. 7, no. 1, pp. 1–6, 2021.

[53] J. Li, P. Zheng, and Y. Yang, "Open-source code for key managament and service framework in decentralized QKD network implementation," https://github.com/QNLab-USTC/Key-Management-and-Service-Framework-for-QKD-Networks, accessed on Dec., 2024.

[54] N. Likhanov, B. Tsybakov, and N. D. Georganas, "Analysis of an ATM buffer with self-similar ("fractal") input traffic," in *Proceedings of INFOCOM'95*, vol. 3. IEEE, 1995, pp. 985–992.

[55] R. K. Jain, D.-M. W. Chiu, W. R. Hawe *et al.*, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, vol. 21, 1984.