# Fast and Secure Routing Algorithms for Quantum Key Distribution Networks

Vishnu B
*Dept. of Electrical Engineering*
Indian Institute of Technology Madras
vishnubeji@gmail.com

Abhishek Sinha
*Dept. of Electrical Engineering*
Indian Institute of Technology Madras
abhishek.sinha@ee.iitm.ac.in

***Abstract*—This paper considers the problem of secure packet routing at the maximum achievable rate in a Quantum key distribution (QKD) network. Assume that a QKD protocol generates symmetric private keys for secure communication over each link in a multi-hop network. The quantum key generation process, which is affected by noise, is assumed to be modeled by a stochastic counting process. Packets are first encrypted with the available quantum keys for each hop and then transmitted on a point-to-point basis over the communication links. A fundamental problem that arises in this setting is to design a secure and capacity-achieving routing policy that accounts for the time-varying availability of the quantum keys for encryption and finite link capacities for transmission. In this paper, by combining the QKD protocol with the Universal Max Weight (UMW) routing policy [1]–[3], we design a new secure throughput-optimal routing policy, called *Tandem Queue Decomposition* (TQD). TQD solves the problem of secure routing efficiently for a wide class of traffic, including unicast, broadcast, and multicast. One of our main contributions in this paper is to show that the problem can be reduced to the usual generalized network flow problem on a *transformed network* without the key availability constraints. Simulation results show that the proposed policy incurs a substantially smaller delay as compared to the state-of-the-art routing and key management policies. The proof of throughput-optimality of the proposed policy makes use of the Lyapunov stability theory along with a careful treatment of the key-storage dynamics.**

***Index Terms*—Quantum Key Distribution, Throughput-optimal routing, Network Algorithms.**

## I. Introduction

Quantum key distribution (QKD) allows remote communication parties to exchange symmetric private keys, whose information-theoretical security is guaranteed by the fundamental principles of quantum mechanics [4]–[6]. The private keys are used for encrypting messages that are communicated over classical channels (*e.g.,* free space or optical fibers). Many QKD protocols are known and already in use, including BB84 [5], E91 [7], and B92 [8]. QKD protocols make use of the physical *Quantum Entanglement* mechanism in an essential way for detecting any possible eavesdropping by an adversarial third party. Once the secret keys have been agreed upon by the peers, the messages between them can be securely encrypted using standard symmetric ciphers, such as One-time pad (OTP) or variants of Advanced Encryption Standard (AES). We emphasize that QKD is used only for distributing the secret keys - the encrypted messages are transmitted exclusively over the classical links. QKD schemes should be contrasted with the ongoing research on Post-Quantum Cryptography (PQC) that are believed to be secure against attack with quantum computers but are lacking sufficient mathematical guarantees for their security properties [9]. Much progress has recently been made in the practical implementations of various QKD schemes [10], [11]. In this paper, we optimize a QKD system similar to the one recently implemented by a team from Oak Ridge and Los Alamos National Labs [12]. See Figure 1 for an illustration of a one-hop QKD system.

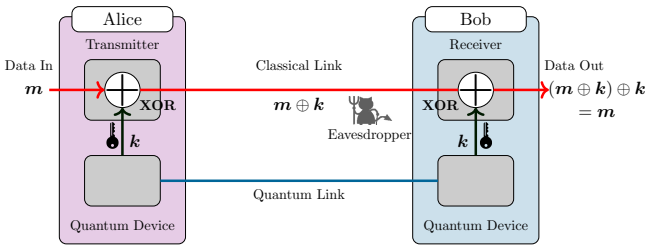


Fig. 1. Depicting a QKD link with the One-Time Pad (OTP) encryption protocol. A sufficiently long symmetric encryption key $\boldsymbol{k}$ is first established between Alice and Bob via the Quantum Link using Quantum Entanglement mechanisms. Next, the message $\boldsymbol{m}$ from Alice to Bob is encrypted at the source by taking XOR of the message with the shared key (bit-by-bit). The encrypted message $\boldsymbol{m} \oplus \boldsymbol{k}$ is then transmitted over the classical link. Upon receiving the encrypted message, Bob securely decrypt XORing it again the same shared secret key as $(\boldsymbol{m} \oplus \boldsymbol{k}) \oplus \boldsymbol{k} = \boldsymbol{m}$. The eavesdropper may try to peek at the message transmitted over the classical link.

Despite having several security advantages, traditional QKD is distance-limited and requires multiple repeaters for long-haul communication. However, these limitations can be mitigated by building QKD networks with stand-alone QKD links. In this paper, we consider the so-called "Trusted Node" setup, where each communication link is assumed to be equipped with a dedicated QKD channel with secure endpoints. Each packet is sequentially encrypted and decrypted along its path by the trusted nodes on each of the intermediate hops [8], [10], [12]–[14]. The transmitted messages on each link must be encrypted to prevent the eavesdropper from compromising the secrecy of the ongoing transmissions. Trusted nodes allow scalable, secure communication, thus overcoming the restrictions imposed by distance-limited pairwise QKD schemes.

From the perspective of resource allocations, each link in a QKD network is characterized by two different resources - (A) the physical link capacity, which remains invariant over time, and (B) the quantum keys currently available for encrypting the link traffic. The latter resource is time-varying and heavily depends on the routing policy (recall that transmitting one message bit using OTP consumes one bit of key). In order to achieve the maximum possible end-to-end throughput, the routing policy must utilize both the resources in an optimal fashion. Throughput-optimal policies for classical networks, such as Back Pressure [15] optimize the routing policy with respect to the link capacities (A) only. The additional constraint stemming from the availability of the keys (B) is unique to the Quantum key distribution Networks, which we address in this paper.

In addition to the regular unicast-type traffic where each packet has a single destination, in this paper, we also consider broadcast and multicast-type traffic, where a single packet needs to be delivered to multiple nodes in the network. Broadcasting and multicasting are essential primitives in tactical military networks where a packet needs to be securely transmitted from a command and control center to multiple terminal nodes. These types of traffic are also common in emergent applications such as video-conferencing and live data-streaming. Our proposed routing algorithm supports any arrival rate within the interior of the capacity region. Furthermore, the proposed algorithm does not need to know either the external packet arrival rates or the quantum key generation rates in advance and works in an online fashion. If the arrival rate vector lies outside the capacity region, suitable admission control mechanisms, such as the one developed in [3], may be used in conjunction with the algorithm developed in this paper.

Our policy, which we name Tandem Queue Decomposition (TQD), works with a virtual network of queues, which are implemented as a vector of counters. The main ingredient of our policy is a new queueing architecture consisting of *two* virtual queues in tandem for each communication link in the network. The reader should compare and contrast this architecture with the original UMW architecture given in [1] that defines *one* virtual queue per link. In our case, the second virtual queue is essential to account for the transmission constraint imposed by the (un)availability of quantum keys. We refer the reader to section III for a detailed description of the construction of the virtual queues. The route of each packet (*e.g.,* a path or tree depending on whether the packet belongs to unicast, multicast, or broadcast flow) is chosen dynamically using a "weighted-shortest-path" computation on the network weighted by the virtual queue lengths. We show that the proposed algorithm is throughput-optimal and demonstrate its efficacy through numerical simulations.

*Related work:* To achieve the network-layer capacity of a multi-hop network, one must use the multi-path routing in an optimal fashion that is commensurate with the external packet arrival rates. In a seminal paper [15], Tassiulas and Ephremides proposed the celebrated *Back-pressure* algorithm, which is proven to be throughput-optimal for unicast traffic. Numerous extensions and enhancements to the basic Back-pressure algorithm have been proposed in the literature for the last thirty years [16]. However, the Back-pressure policy is specific to unicast flows only, and hence, it does not support broadcast or multicast traffic [17]. Using the Back-pressure algorithm as a building block, the paper [18] proposes quantum key management and unicast routing policy to maximize the utility in a QKD network. In addition to being limited to unicast flows only, a major technical limitation of the scheme of [18] is that, in order to stabilize the data queues, the authors place an artificial upper bound on the number of keys a node can have at its disposal (see Lemma 1 of [18]). However, unlike the data packets in transit, the abundance of quantum keys is always desirable as they can be used to encrypt more data packets. Hence, the utility achieved by the algorithm in [18] could be sub-optimal.

Building upon our earlier work on the Universal Max-Weight algorithm [1], in this paper, we propose an efficient universal routing policy (TQD) that does not limit the number of keys in a node yet achieves the full throughput region of the network. Our policy supports a wide range of traffic types, including unicast, broadcast, and multicast. Furthermore, since TQD admits loop-free routing, it offers a better delay performance than [18] for unicast flows. To achieve this, TQD uses the UMW policy on a transformed network with twice the number of edges compared to the original network.

The rest of the paper is organized as follows: In section (II), we discuss the system model and formulate the problem. In section (III), we give a brief overview of the proposed TQD policy and the structure and dynamics of the virtual queues on which TQD is based. In section (IV), we prove its stability property in the multi-hop physical network. In section (V), we present some illustrative numerical simulation results before concluding the paper in section (VI).

## II. System Model and Problem Formulation

### A. Network Model

We consider a network with an arbitrary topology, represented by a graph $\mathcal{G}(V, E)$, where $V$ denotes the set of nodes ($|V| = n$) and $E$ denotes the set of edges ($|E| = m$). The edges could be either directed or undirected. Time evolves in discrete slots. Each edge in the network encompasses two types of links - a physical link and a QKD link. The capacity of the physical link $e$ is $\gamma_e$, *i.e.,* it can transmit $\gamma_e$ number of encrypted packets per slot. The QKD links are used for symmetric quantum key agreement between the nodes and *not* for the actual data transfer, which takes place over the physical links. Although the network topology is assumed to be static in this paper, our proposed policy works even in the case of a time-varying network. Furthermore, all of our results can be straightforwardly generalized to networks with scheduling constraints (*e.g.,* wireless networks).

### B. Quantum Keys - Generation, Distribution, and Utilization

The keys generated by the quantum entanglement mechanism between two trusted nodes are shared via the overlay QKD

channels. The generated keys are stored in *key banks*, which are typically implemented with text files on both sides of the links [12]. Note that the key banks are different from the data queues; while the data queues hold physical data packets, the key banks store private symmetric quantum keys, which are used for encrypting or decrypting the data packets before each transmission (see Figure 1). Due to noise originating from quantum decoherence, the amount of usable quantum keys generated per slot varies randomly. Let $K_e(t)$ be the number of keys generated over the QKD link $e$ in the time slot $t$. In this paper, we assume that $\{K_e(t)\}_{t\geq 1}$ is an i.i.d. stochastic process with $\mathbb{E}(K_e(t)) = \eta_e$ such that $K_e(t) \leq K_{\max}, \forall t, e$ for some finite constant $K_{\max}$. Without any loss of generality, we normalize the unit of the key generation so that one unit of key encodes precisely one data packet for a given encryption standard. Due to the technological and physical challenges arising from the entanglement generation and quantum decoherence [19], the quantum key generation is usually the bottleneck for information transmission [20]. Since the abundance of encryption keys is always desirable, we do not impose any hard upper limit on the size of the key banks. Thus, unlike the paper [18], we do not require the key banks to be stable. Our objective is to design a policy that stabilizes only the data queues for any arrival rate within the capacity region (see Definition 3).

### C. Data Traffic Model

We consider a generalized traffic model, where a packet arriving at a source node $s$ can have either a single destination (Unicast), or multiple destinations (Multicast) [1]. A special case of Multicast traffic is Broadcast, where the incoming packet to a node is required to be delivered to *all* other nodes in the network. Formally, we categorize the incoming packets into multiple classes $\mathcal{C}$ depending on its source ($s^{(c)}$) and the set of destination(s) ($\mathcal{D}^{(c)}$). Packets belonging to the class $c$ is assumed to arrive at the source at rate $\lambda^c$ i.i.d. at every slot. In other words, if $A^{(c)}(t)$ denotes the number of external packets from class $c$ that arrives at the source $s^{(c)}$, we have $\mathbb{E}A^{(c)}(t) = \lambda^{(c)}, \forall c \in \mathcal{C}$. The arrival rate vector $\boldsymbol{\lambda}$ is obtained by concatenating the arrival rates from each class, *i.e.,* $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_{|\mathcal{C}|})$. We also assume that the total number of new packet arrivals to the entire network at any time slot is upper bounded by a finite constant $A_{\max}$.

### D. Policy Space

An admissible policy for this problem is responsible for the following tasks - (1) selecting the routes for each packet based on their traffic class and possibly duplicate the packets as necessary (in the case of broadcast and multicast traffic), (2) encrypting the link traffic with the available keys, and (3) forwarding the encrypted packets over the communication links. Note that a data packet can be forwarded over a link only if sufficiently many quantum keys are available for encryption. Otherwise, the policy must wait until the keys are generated. The set of all admissible policies is denoted by $\Pi$.

We say that a policy $\pi \in \Pi$ *securely* supports an arrival rate vector $\boldsymbol{\lambda}$ if under the action of the policy $\pi$, the destination node(s) of class $c$ receive(s) encrypted class $c$ packets at the rate $\lambda^{(c)}$, $\forall c \in C$. Formally, let $R^{(c)}(t)$ denote the total number of encrypted class $c$ packets commonly received by the destination node(s) $\mathcal{D}^{(c)}$ under the action of the policy $\pi$ up to time $t$. We now make the following definitions.

**Definition 1 (Policy Securely Supporting an Arrival Rate Vector $\boldsymbol{\lambda}$).** *A policy $\pi \in \Pi$ is said to securely support an arrival rate vector $\boldsymbol{\lambda}$ if*

$$\liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} = \lambda^{(c)}, \quad \forall c \in \mathcal{C}, \quad \textit{w.p. } 1$$

**Definition 2 (Stability Region of a Policy).** *The stability region $\Lambda_\pi(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ of an admissible policy $\pi$ is defined to be the set of all arrival rate vectors securely supported by the policy $\pi$,* i.e.,

$$\boldsymbol{\Lambda_\pi}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma}) \overset{\text{(def)}}{=} \{\boldsymbol{\lambda} \in \mathbb{R}_+^{|\mathcal{C}|} : \pi \text{ securely supports } \boldsymbol{\lambda}\},$$

In the above definition, we have made the dependence of the stability region with the network topology ($\mathcal{G}$), key-generation rates ($\boldsymbol{\eta}$), and the link capacities ($\boldsymbol{\gamma}$) explicit. The secure capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ is defined to be the set of all arrival rate vectors supported by an admissible policy. Formally,

**Definition 3 (Secure Capacity Region of a Network).** *The secure capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ of a network is defined to be the set of all supportable rates, i.e.,*

$$\boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma}) = \bigcup_{\pi\in\Pi} \boldsymbol{\Lambda_\pi}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma}).$$

Finally, we define the notion of a *secure* Throughput-Optimal policy, which generalizes the notion of Throughput-Optimal policies as given in [15].

**Definition 4 (Secure Throughput-Optimal Policy).** *A secure throughput-optimal policy is an admissible policy $\pi^* \in \Pi$, that supports any arrival rate $\boldsymbol{\lambda}$ in the interior of the secure capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$.*

From the above definition, it is unclear whether a secure throughput-optimal policy exists as two different rate vectors in the secure capacity region might not be achieved by the same admissible policy. One of the major contributions of the paper is to show that a secure throughput-optimal policy exists and can be efficiently implemented.

### E. Characterization of the Secure Capacity Region

Let $\mathcal{G}_{\boldsymbol{\omega}}$ be a capacitated version of the given network such that the link capacity $\omega_e$ for the edge $e$ is defined as follows:

$$\omega_e = \min(\gamma_e, \eta_e), \qquad \forall e \in E.$$

In Theorem 1 below, we show that the capacity region of the network is given by the generalized multi-commodity flow region of the capacitated graph $\mathcal{G}_{\boldsymbol{\omega}}$. One direction of this result is quite intuitive; the long-term rate of encrypted packet flow over an edge $e$ is limited by the quantum key

generation rates and the capacity of the communication link $e$. Consider an arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$. By definition, there exists an admissible policy $\pi \in \Pi$ that supports the arrival rate $\boldsymbol{\lambda}$. Upon taking a long-term time-average over the actions of the policy $\pi$, it is evident that we can obtain a *randomized* flow-decomposition on $\mathcal{G}_\omega$ such that none of the edges are overloaded. In other words, for every $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\gamma}, \boldsymbol{\eta})$, there exist a non-negative scalar $\lambda_i^{(c)}$, associated with the $i^{\text{th}}$ admissible route $T_i^{(c)} \in \mathcal{T}^{(c)}, \forall i, c$, such that

$$\lambda^{(c)} = \sum_{i:T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)}, \tag{1}$$

$$\lambda_e \overset{\text{(def.)}}{=} \sum_{\substack{(i,c):e \in T_i^{(c)}, \\ T_i^{(c)} \in \mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \omega_e, \quad \forall e \in E. \tag{2}$$

Eqn. (1) shows that there exists such a valid flow decomposition across the routes. The inequality in (2) states that no edge in $\mathcal{G}_{\boldsymbol{\omega}}$ is overloaded. To formally state our result we need the following definition for the feasible flow region $\overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}}$ of $\mathcal{G}_{\boldsymbol{\omega}}$.

**Definition 5.** *$\overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}}$ is defined as the set of all arrival vectors $\boldsymbol{\lambda} \in \mathbb{R}_+^{|\mathcal{C}|}$ for which there exists a non-negative flow decomposition $\{\lambda_i^{(c)}\}$ such that Eqns. (1) and (2) are satisfied.*

Let $\text{int}(\cdot)$ denote the interior of a subset of an $n$-dimensional Euclidean space. The following theorem characterizes the secure capacity region of a network.

**Theorem 1** (Characterization of the Capacity region)**.** *The network-layer secured capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ is identical to the set $\overline{\boldsymbol{\Lambda}}$, up to the boundary, i.e.,*

1) **[Converse]** $\boldsymbol{\Lambda} \subseteq \overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}}$.
2) **[Achievability]** $\text{int}(\overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}}) \subseteq \boldsymbol{\Lambda}$ *and there exists an admissible policy which achieves any rate in* $\text{int}(\overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}})$.

The proof of the converse (i) is given in Appendix A. The achievability result in part (ii) is more challenging. We establish the achievability by designing an efficient policy, called TQD, that supports all rates within the set $\text{int}(\overline{\boldsymbol{\Lambda}})$.

## III. A Secure Throughput-Optimal Policy

In this section, we describe a policy that achieves the entire secure capacity region of a network. The presence of the key availability constraints makes this problem more challenging than the vanilla universal network flow problem considered in [1]. We solve this problem using a novel yet straightforward Tandem Queue Decomposition (TQD) method that reduces the current problem to an instance of the universal flow problem.

### A. The Tandem Queue Decomposition Method (TQD)

To enforce the constraint that only encrypted packets can be transmitted over the physical links, we first construct a *transformed network* where each edge is split into two edges in tandem, each containing one queue. The first edge, which is internal to the nodes, corresponds to the process of encrypting the packets with the available quantum keys. The second edge corresponds to the process of transmitting the encrypted packets over the physical links. We illustrate the construction of the transformed network via the following example.

*Example:* Consider an edge $(A, B)$ that connects node $A$ to node $B$ as shown in Figure 2. Assume that, the quantum keys are generated for the link $(A, B)$ at the rate $\eta_{AB}$ and the communication link can transmit packets at the rate of $\gamma_{AB}$ packets per second. In the transformed network, we replace the edge $(A, B)$ by introducing two internal nodes $a_1$ and $a_2$, an *internal* edge $(a_1, a_2)$ connecting them, and an *external* edge $(a_2, b_1)$. The queue $X_{AB}$, corresponding to the internal edge $(a_1, a_2)$, holds the set of *unencrypted* packets that are waiting for the keys to become available. The queue $Y_{AB}$, corresponding to the external edge $(a_2, b_1)$, holds the set of *encrypted* packets that are waiting to cross the link $AB$. Similar decompositions are performed for each link in the network.

It is easy to see that the above transformation does not change the capacity region of the network. Hence, it is enough to design a throughput-optimal policy for the transformed network. In the following, we use the Universal Max-Weight policy [1] on the transformed network for achieving this goal.

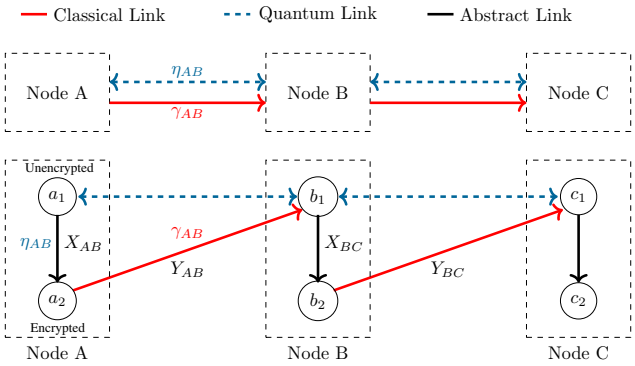


Fig. 2. Consider a data packet moving from Node $A$ to Node $C$ via Node $B$. The upper part of the schematic shows the physical connections between the nodes, and the bottom part denotes its algorithmic abstraction after the Tandem Queue Decomposition. The link between two trusted nodes $A$ and $B$ consists of the classical data link for packet transfer, the quantum link for mutual key agreement, and an abstract link to an intermediate node where the encrypted data is stored before sending it via the communication link.

### B. Precedence Constraints and the Virtual Queueing Process

Due to the precedence constraints, a packet, which is routed along the route $T = e_1 - e_2 - ... - e_n$, reaches the $j^{\text{th}}$ link $e_j$ only after crossing the links located before $e_j$. This follows from the principle of causality applied to multi-hop networks. Similar to the UMW policy, we first relax this constraint to obtain a single-hop *virtual network*, which is used for deciding the routes for the incoming packets.

Similar to UMW, we define a $2m$-dimensional virtual queueing process $\tilde{\boldsymbol{Q}}(t) := \{\tilde{\boldsymbol{X}}(t), \tilde{\boldsymbol{Y}}(t)\}$. We associate one virtual queue corresponding to each edge in the transformed network. Hence, $\tilde{\boldsymbol{X}}$ corresponds to the virtual queues consisting of the unencrypted packets waiting for keys, and $\tilde{\boldsymbol{Y}}$ corresponds to the virtual queues consisting of the encrypted packets waiting to be transmitted over the communication links in the virtual

network. We emphasize that virtual queues correspond to a state vector that follows simplified queueing dynamics without precedence constraints [1].

> **Note:** In the physical system, there are two tandem queues, with $X_e$ followed by $Y_e$ for each edge $e \in E$ as shown in Figure 2. The corresponding virtual queues are denoted by $\tilde{X}_e$ and $\tilde{Y}_e$. The packets in $X_e$ at slot $t$ get encrypted with the available quantum keys in the key bank and then forwarded to the physical queue $Y_e$. From $Y_e$, the packets get transmitted to the next link $e'$ and get queued at $X_{e'}$.

The TQD policy first decides a route $T^{(c)}(t) \in \mathcal{T}^{(c)}$ for a class $c$ packet immediately upon its arrival. If we denote the links on a prescribed route $T^{(c)}(t)$ by $\{e_i | i = 1, 2, \ldots, k\}$, the incoming packet induces a virtual packet arrival at slot $t$ simultaneously at each of the virtual queues $\{\tilde{X}_{e_i} | i = 1, 2, \ldots, k\}$ and $\{\tilde{Y}_{e_i} | i = 1, 2, \ldots, k\}$. By foregoing the precedence constraints, any packet present in the virtual queues can be serviced at the same time. Thus the number of packet arrivals $A_e^{\pi}(t)$ to both the virtual queues $\tilde{Q}_e = \{\tilde{X}_e, \tilde{Y}_e\}$ at time $t$ under the action of a policy $\pi$ can be expressed as:

$$A_e^{\pi}(t) = \sum_{c \in \mathcal{C}} A^{(c)}(t) \mathbb{1}\big(e \in T^{(c)}(t)\big), \quad \forall e \in E. \tag{3}$$

The symmetric quantum key pairs generated at slot $t$ are stored into the key banks on both sides of the edge $e$. The keys that are not utilized for encryption in the current slot are stored for use in the future. Let $\kappa_e(t)$ denote the number of keys available for encrypting the packets crossing the edge $e$ at slot $t$ and $k_e(t)$ denote the number of unused keys in the key bank available from the previous rounds. Hence, $\kappa_e(t) = K_e(t) + k_e(t)$, where we recall that $K_e(t)$ is the number of *new* keys generated by the QKD link $e$ at slot $t$. Note that the process $\{\boldsymbol{\kappa}(t)\}_{t \geq 1}$ is highly *correlated* with the routing policy.

Thus, the one-step evolution (Lindley recursion) of the virtual queue processes $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{Y}}$ can be written as:

$$\tilde{X}_e(t+1) = \big(\tilde{X}_e(t) + A_e^{\pi}(t) - \kappa_e(t)\big)^+, \quad \forall e \in E \tag{4}$$

$$\tilde{Y}_e(t+1) = \big(\tilde{Y}_e(t) + A_e^{\pi}(t) - \gamma_e\big)^+, \quad \forall e \in E. \tag{5}$$

With the above description of the queueing architecture in place, we now present our proposed capacity-achieving routing policy in Algorithm 1. We note down a few salient feature of the algorithm:

1) The routing policy is oblivious to the arrival rates $\boldsymbol{\lambda}$, the key generation rates $\boldsymbol{\eta}$ and the capacities of the links $\boldsymbol{\gamma}$.
2) The shortest path computations depend on the virtual queue lengths through the sum of the encrypted and unencrypted queues in each link and not on the individual virtual queue lengths.

In the following section, we show that the proposed policy stabilizes the virtual and the physical queues for all arrival rates within the interior of the capacity region.

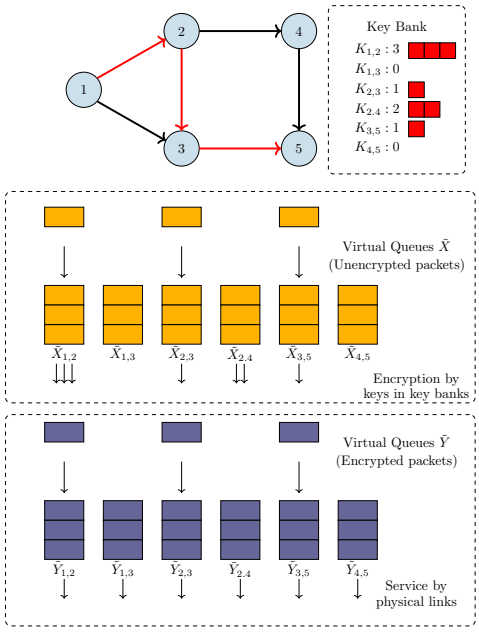


Fig. 3. Illustration of the virtual queue system for the 5-node network $\mathcal{G}$. When a packet arrives at source 1 with desination 5, and given the assigned route $\mathcal{T}_p = \{\{1,2\},\{2,3\},\{3,5\}\}$, the following queue updations occur. The packet is counted simultaneously as an arrival to the virtual data queues $\tilde{X}_{12}, \tilde{X}_{23}, \tilde{X}_{35}, \tilde{Y}_{12}, \tilde{Y}_{23}, \tilde{Y}_{35}$ *at the same slot*. Following this, we reserve the quantum keys at simultaneously at the same slot, thereby reducing the number of quantum keys in those virtual queues. The physical packet reaches these edges only at a later time slot.

## IV. Derivation of a Stabilizing Routing Policy for the Virtual Network

Due to the packet transmission constraint arising from the availability of the quantum keys in the key banks, our proof of strong stability differs from the proof given in [1]. To derive a stabilizing policy for the virtual network, we consider the following quadratic Lyapunov function $L(\tilde{Q}(t))$ defined in terms of the virtual queue lengths:

$$L(\tilde{Q}(t)) = \sum_{e \in E} \big(\tilde{X}_e^2(t) + \tilde{Y}_e^2(t)\big). \tag{6}$$

The Lyapunov function (6) defined above should be contrasted with the Lyapunov function considered in Eqn. (9) of [18], which includes the number of available keys as the second term in the quadratic. Thus, any drift minimizing policy for their Lyapunov function implicitly limits the number of usable keys in the key banks as well, which is not practically desirable. From the one-step dynamics given in Eqn. (4), we have:

$$\begin{aligned}\tilde{X}_e(t+1)^2 \leq{}& \tilde{X}_e(t)^2 + A_e^{\pi 2}(t) + \kappa_e(t)^2 + 2\tilde{X}_e(t)A_e^{\pi}(t) \\ &- 2\tilde{X}_e(t)\kappa_e(t) - 2A_e^{\pi}(t)\kappa_e(t), \\ \tilde{Y}_e(t+1)^2 \leq{}& \tilde{Y}_e(t)^2 + A_e^{\pi 2}(t) + \gamma_e^2 + 2\tilde{Y}_e(t)A_e^{\pi}(t) \\ &- 2\tilde{Y}_e(t)\gamma_e - 2A_e^{\pi}(t)\gamma_e.\end{aligned}$$

**Algorithm 1** Tandem Queue Decomposition (TQD) algorithm at slot $t$ for the Generalized Flow Problem

**Require:** Graph $\mathcal{G}(V,E)$, Virtual Queue lengths $\{\tilde{X}_e(t), e \in E\}$ and $\{\tilde{Y}_e(t), e \in E\}$ at the slot $t$

1: **[Edge-Weight Assignment]** Assign each edge of the original graph $e \in E$ a weight $W_e(t)$ equal to $\tilde{X}_e(t) + \tilde{Y}_e(t)$, i.e
$$\boldsymbol{W}(t) \leftarrow \tilde{\boldsymbol{X}}(t) + \tilde{\boldsymbol{Y}}(t).$$
2: **[Route Assignment]** Compute a Minimum-Weight Route $T^{(c)}(t) \in \mathcal{T}^{(c)}(t)$ for a class $c$ incoming packet in the weighted graph $\mathcal{G}(V,E)$.
3: **[Key Generation]** Generate symmetric private keys for every edge $e$ via QKD and store them in the key banks.
4: **[Encryption]** Encrypt the data packets waiting in physical queue $X_e$ with the available keys in the key bank and move the encrypted packets to the downstream queue $Y_e$ for every edge $e$.
5: **[Packet Forwarding]** Forward the encrypted physical packets from the queue $Y_e$ to the queue $X_{e'}$ for every edge $e$ according to some packet scheduling policy (ENTO, FIFO etc). Here $e'$ is the next edge in the assigned route of a packet.
6: **[Decryption]** Decrypt the data packets received at physical queue $X_e$ for every edge $e$ using the symmetric key generated earlier via the QKD process.
7: **[Queue Counter Updation]** Update the virtual key queues and virtual data queues assuming a precedence-relaxed system, i.e.,
$$\tilde{X}_e(t+1) \leftarrow \big(\tilde{X}_e(t) + A_e^{\pi}(t) - \kappa_e(t)\big)^+, \quad \forall e \in E$$
$$\tilde{Y}_e(t+1) \leftarrow \big(\tilde{Y}_e(t) + A_e^{\pi}(t) - \gamma_e\big)^+, \quad \forall e \in E.$$

Since $\tilde{X}_e(t) \geq 0, \kappa_e(t) \geq 0$, $A_e^{\pi}(t) \geq 0$ and $\gamma_e \geq 0$, we can write:

$$\tilde{X}_e(t+1)^2 - \tilde{X}_e(t)^2 \leq A_e^{\pi 2}(t) + \kappa_e(t)^2 + 2\tilde{X}_e(t)A_e^{\pi}(t) - 2\tilde{X}_e(t)\kappa_e(t), \quad (7)$$

$$\tilde{Y}_e(t+1)^2 - \tilde{Y}_e(t)^2 \leq A_e^{\pi 2}(t) + \gamma_e^2 + 2\tilde{Y}_e(t)A_e^{\pi}(t) - 2\tilde{Y}_e(t)\gamma_e. \quad (8)$$

Next, observe that $\tilde{X}_e(t)k_e(t) = 0$. This can be argued as follows. Since all currently available keys are used for encryption, if there are packets in the queue waiting to be encrypted (*i.e.,* if $\tilde{X}_e(t) > 0$), there cannot be any unused keys from the previous round (*i.e.,* $k_e(t) = 0$.) Since $\kappa_e(t) = k_e(t) + K_e(t)$, for $\tilde{X}_e(t) > 0$, we can rewrite the inequality (7) as:

$$\tilde{X}_e^2(t+1) - \tilde{X}_e^2(t) \leq A_e^{\pi 2}(t) + K_e^2(t) + 2\tilde{X}_e(t)A_e^{\pi}(t) - 2\tilde{X}_e(t)K_e(t). \quad (9)$$

On the other hand, if $\tilde{X}_e(t) = 0$, trivially we have $\tilde{X}_e(t+1) \leq A_e^{\pi}(t)$. Thus, even in this case, the bound in Eqn. (9) continues to hold. Hence, from Eqn. (9) and (8), the expected one-step Lyapunov drift $\Delta^{\pi}(t)$, conditioned on the current virtual queue lengths $\tilde{Q}(t)$, under the operation of any admissible policy $\pi \in \Pi$ may be upper bounded as:

$$\begin{aligned}\Delta^{\pi}(t) &\equiv \mathbb{E}\Big(L(\tilde{Q}(t+1)) - L(\tilde{Q}(t))|\tilde{\boldsymbol{Q}}(\boldsymbol{t})\Big) \\ &\leq B + 2\sum_{e\in E}\big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big)\mathbb{E}\big(A_e^{\pi}(t)|\tilde{\boldsymbol{Q}}(\boldsymbol{t})\big) \\ &\quad - 2\sum_{e\in E}\tilde{X}_e(t)\eta_e - 2\sum_{e\in E}\tilde{Y}_e(t)\gamma_e, \end{aligned} \quad (10)$$

where $B$ is a finite constant that depends on the upper bounds of the packet arrival and quantum key generation rates. Note that, in Eqn. (10), we have used the fact that

$$\mathbb{E}\big(K_e(t)|\tilde{\boldsymbol{Q}}(t)\big) = \mathbb{E}(K_e(t)) = \eta_e.$$

### *A. A Drift Minimizing Routing Policy:*

We now design a routing policy which minimizes the upper bound for the one-step Lyapunov drift as given by the RHS of Eqn. (10). By inspecting the terms, it is clear that the policy must choose the route for each packets to minimize the following routing cost:

$$\text{RoutingCost}^{\pi} = \sum_{e\in E}\big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big)A_e^{\pi}(t).$$

Using Eqn (3), we can express this cost as:

$$\text{RoutingCost}^{\pi} = \sum_{c\in C}A^{(c)}(t)\sum_{e\in E}\big(\tilde{X}_e(t)+\tilde{Y}_e(t)\big)\mathbb{1}(e \in T^{(c)}(t)),$$

where $T^{(c)}(t) \in \mathcal{T}^{(c)}(t)$ and $\mathcal{T}^{(c)}(t)$ is the set of all admissible routes for the packets belonging to the traffic class $c$. Decomposing the above cost function into distinct traffic classes, we see that the drift minimizing policy chooses routes for the packets in class $c$ at time $t$ by solving the following combinatorial optimization problem:

$$T_{\text{opt}}^{(c)}(t) \in \underset{T^{(c)}\in\mathcal{T}^{(c)}(t)}{\arg\min}\sum_{e\in E}\big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big)\mathbb{1}(e \in T^{(c)}) \quad (11)$$

Let $\tilde{Z}_e(t) \equiv \tilde{X}_e(t) + \tilde{Y}_e(t)$ be the sum of the lengths of the virtual queues (consisting of both unencrypted and encrypted packets) for the edge $e$ at time $t$. Now consider an edge-weighted version of the graph $G$, where the weight of the edge $e$ is taken to be $\tilde{Z}_e(t)$. For different traffic types, the optimal route for each packet is chosen as follows:

- **Unicast:** The shortest $s^{(c)} - t^{(c)}$ path in the weighted-graph.
- **Broadcast:** The minimum-weight spanning tree(MST) with root $s^{(c)}$, in the weighted-graph.
- **Multicast:** The minimum-weight Steiner tree with root $s^{(c)}$ and covering all destinations $\mathcal{D}^{(c)}$ in the weighted-graph.
- **Anycast:** The shortest of the $k$ shortest $s^{(c)} - t_i^{(c)}, 1 \leq i \leq k$ paths in the weighted-graph.

For routing multicast traffic, we may use an efficient approximation algorithm for the Min-weight Steiner tree problem (such as the one described in [21]), as solving the problem optimally is **NP-hard** for arbitrary graphs. For all other traffic classes, there exist efficient and standard algorithms that may be directly used [22].

### *B. Strong Stability of the Virtual Queues*

**Theorem 2.** *Under the TQD routing policy, the virtual queue process $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 0}$ is strongly stable for any arrival rate vector $\boldsymbol{\lambda} \in$ int$(\overline{\boldsymbol{\Lambda}})$,* i.e.,

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{e\in E} \mathbb{E}(\tilde{X}_e(t) + \tilde{Y}_e(t)) < \infty$$

*Proof:* Consider an arrival rate vector $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$. From Eqns. (1) and (2), it follows that there exists a scalar $\epsilon > 0$ such that we can decompose the total arrival for each class $c \in \mathcal{C}$ into a finite number of routes, such that

$$\lambda_e = \sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \omega_e - \epsilon, \quad \forall e \in E. \tag{12}$$

We now define an auxiliary stationary randomized routing policy $\pi_{\textbf{RAND}} \in \Pi$ such that the policy $\pi_{\textbf{RAND}}$ assigns an incoming packet from class $c$ the route $T_i^{(c)} \in \mathcal{T}^{(c)}(t)$ with probability $\frac{\lambda_i^{(c)}}{\lambda^{(c)}}, \forall i, c$. Hence, it follows that the expected number of packets that is routed along a path (or tree) that includes the edge $e$ is given by:

$$\mathbb{E}(A_e^{\pi_{\textbf{RAND}}}(t)) = \lambda_e = \sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \lambda_i^{(c)}, \quad \forall e \in E. \tag{13}$$

Since the TQD policy minimizes the drift expression in Eqn. (10) among the set of all feasible routing policies $\pi \in \Pi$, by comparing it with the randomized policy $\pi_{\textbf{RAND}}$, we can write:

$$\begin{aligned}\Delta^{\pi_{\textbf{TQD}}}(t) \leq B + 2\sum_{e\in E} \big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big)\mathbb{E}\Big(A_e^{\pi_{\textbf{RAND}}}(t)|\tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t})\Big) \\ - 2\sum_{e\in E} \tilde{X}_e(t)\eta_e - 2\sum_{e\in E} \tilde{Y}_e(t)\gamma_e. \end{aligned} \tag{14}$$

Using the fact that Randomized policy is memoryless, and hence, independent of the virtual queue lengths $\tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t})$, the above drift inequality simplifies to:

$$\begin{aligned}\Delta^{\pi_{\textbf{TQD}}}(t) &\leq B + 2\sum_{e\in E} \Big((\lambda_e - \gamma_e)\tilde{X}_e(t) + (\lambda_e - \eta_e)\tilde{Y}_e(t)\Big) \\ &\leq B + 2\sum_{e\in E} \Big((\lambda_e - \min(\gamma_e,\eta_e))\tilde{X}_e(t) \\ &\qquad + (\lambda_e - \min(\gamma_e,\eta_e))\tilde{Y}_e(t)\Big) \\ &= B + 2\sum_{e\in E} (\lambda_e - \omega_e)\big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big) \\ &\leq B - 2\epsilon \sum_{e\in E} \big(\tilde{X}_e(t) + \tilde{Y}_e(t)\big),\end{aligned}$$

where we have used the inequality from Eqn. (12). Taking expectation of both sides w.r.t. the virtual queue lengths $\tilde{Q}(t)$, we bound the expected drift at slot $t$ as:

$$\mathbb{E}L(\tilde{\boldsymbol{Q}}(t+1)) - \mathbb{E}L(\tilde{\boldsymbol{Q}}(t)) \leq B - 2\epsilon \sum_{e\in E} \mathbb{E}(\tilde{X}_e(t) + \tilde{Y}_e(t)).$$

Upon summing the above inequality from $t = 0$ to $T-1$, dividing both sides by $T$ and upon realizing that $L(\tilde{\boldsymbol{Q}}(0)) = 0$ we have:

$$\frac{\mathbb{E}L(\tilde{Q}(T))}{T} + \frac{1}{T}\sum_{t=0}^{T-1}\sum_{e\in E} \mathbb{E}(\tilde{X}_e(t) + \tilde{Y}_e(t)) \leq \frac{B}{2\epsilon}. \tag{15}$$

Finally, using the fact that $L(\tilde{\boldsymbol{Q}}(T)) \geq 0$ and $L(\tilde{\boldsymbol{Q}}(0)) = 0$, we get

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{e\in E} \mathbb{E}(\tilde{X}_e(t) + \tilde{Y}_e(t)) \leq \frac{B}{2\epsilon}.$$

Taking $\limsup$ on both sides we get that

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{e\in E} \mathbb{E}(\tilde{X}_e(t) + \tilde{Y}_e(t)) < \infty,$$

which shows that the virtual queue processes $\{\tilde{\boldsymbol{X}}(t)\}_{t\geq 1}$ and $\{\tilde{\boldsymbol{Y}}(t)\}_{t\geq 1}$ are strongly stable. ■

*Observation:* It is clear that the proof of Theorem 2 goes through even when we do not store the keys from the past, *i.e.* the freshly-generated keys at each slot are used for encrypting the packets for that slot only and the excess keys (if any) are discarded at the end of the slots ($k_e(t) = 0, \forall t, e$). This is obviously a practically wasteful way of operating the system, but it does not affect the throughput-optimality of the TQD policy. Moreover, one advantage of this scheme is that we can now operate the system with *zero-sized* key banks and discard stale (and potentially unsecured) keys without losing capacity. See section V for a numerical evaluation of its performance.

### *C. Stability of the Physical Queues*

The physical queues obey precedence constraints and have a dynamics different from the virtual queues. In the following, we argue that the physical queues $\{\boldsymbol{X}(t)\}_{t\geq 1}$ and $\{\boldsymbol{Y}(t)\}_{t\geq 1}$ are also stable.

*a) Stability of the $\{\boldsymbol{X}(t)\}_{t\geq 1}$ process:* Note that the number of keys generated at each slot for serving the virtual queue $\tilde{X}_e(t)$ and the physical queue $X_e(t)$ are identical for all edges $e \in E$ and time slot $t$. Since the excess keys are indefinitely stored in the key banks and by design, the packet arrivals are counted the virtual queue of unencrypted packets $\tilde{X}_e$ *before* they arrive in the corresponding physical queue $X_e$, it readily follows that

$$X_e(t) \leq \tilde{X}_e(t), \ \forall e, t. \text{ a.s.} \tag{16}$$

Hence, from Theorem IV-B, it directly follows that the physical queues consisting of unencrypted packets are strongly stable.

*b) Stability of the $\{\boldsymbol{Y}(t)\}_{t\geq 1}$ process:* Since, unlike the upstream queues $X_e(t)$'s, the cumulative unused services of the links can not be stored for later use, it is not possible to derive a simple pairwise inequality similar to (16) for the downstream queues $Y_e(t), \forall e \in E$. Using the fact that the virtual queue processes $\{\tilde{\boldsymbol{X}}(t)\}_{t\geq 1}$ and $\{\tilde{\boldsymbol{Y}}(t)\}_{t\geq 1}$ are non-negative, and $L(\tilde{\boldsymbol{Q}}(t)) \geq \tilde{Y}_e^2(t), \forall e \in E$, (viz. Eqn. (6)) from Eqn. (15) we have $\mathbb{E}\tilde{Y}_e^2(T) \leq \frac{BT}{2\epsilon}, \forall T \geq 1$. Furthermore, since the number of packet arrivals at a slot and the link capacities are bounded, we see that the conditions of Lemma 3.2 of [23] are satisfied. Hence, under the TQD policy, we have for any $\lambda \in \text{int}(\overline{\boldsymbol{\Lambda}}_{\boldsymbol{\omega}})$:

$$\lim_{t\to\infty} \frac{\tilde{Y}_e(t)}{t} = 0, \quad \forall e \in E, \qquad \text{a.s.} \tag{17}$$

Next, using an appropriate packet scheduling policy for the encrypted packets for the outgoing physical links (*e.g.,* the Nearest to Origin policy [24]), it can be shown that the rate stability (17) of the virtual queues implies the rate stability of the physical queues for the encrypted packets as well, *i.e.,*

$$\lim_{t\to\infty} \frac{\sum_{e\in E} Y_e(t)}{t} = 0, \qquad \text{w.p. } 1. \tag{18}$$

We refer the readers to [1] Theorem 3 for a detailed proof using adversarial queueing theory. From the above, it immediately follows that the TQD policy is throughput-optimal. We give a formal proof of this result in Appendix B.

## V. Simulation Results

### A. TQD with Unicast traffic

We simulate the TQD policy on the network shown in Figure 4. All physical links have unit capacity with the quantum key-generation rates as indicated in the Figure. We consider two unicast flows from the sources $s_1$, $s_2$ to the destinations $t_1$, $t_2$ with the arrival rates $\lambda_1$ and $\lambda_2$ respectively. The packet arrivals and key-generation processes are assumed to be Poisson.

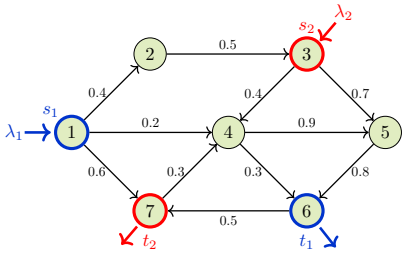


Fig. 4. Network topology used for unicast simulation.

In our simulations, we compare the performance of the proposed *Tandem Queue Decomposition* policy (with and without the key-storage) with the Back-Pressure-based routing policy proposed in [18]. Let $\lambda_1^* = 0.6$ and $\lambda_2^* = 0.5$. It can be shown from an LP formulation that $\lambda_1 + \lambda_2 \leq \lambda_1^* + \lambda_2^* = 1.1$, for any feasible rate pair $(\lambda_1, \lambda_2)$. Figure 5 shows the variation of average physical queue-lengths when the arrival rates are varied as $(\lambda_1, \lambda_2) = (\rho\lambda_1^*, \rho\lambda_2^*)$, for $0 < \rho < 1$. The physical queue lengths are averaged over 1000 sample paths, each run for 10000 time slots. Plot 5 shows that the TQD policy with key storage performs the best, followed by the Back-Pressure policy. The TQD policy without key storage is throughput-optimal, but its delay performance is poor due to the discarding of excess quantum keys.

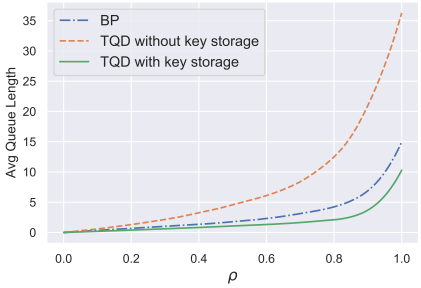


Fig. 5. Performance comparison between the **TQD** policy (with and without key storage) and the **Back pressure** policy in the unicast setting

### B. TQD with Broadcast traffic

In our broadcast simulations, we consider a $3 \times 3$ grid network with the source located in the upper-leftmost corner. Each edge has a uniform key generation rate $\eta_e = 0.5$ and unit capacity. It can be shown that the broadcast capacity of the network is 0.5 [2]. Figure 6 compares the performance of two variants of the TQD policy. We see that both policies are capacity-achieving, yet the TQD policy with key storage has a far better performance compared to its no-key-storage variant.

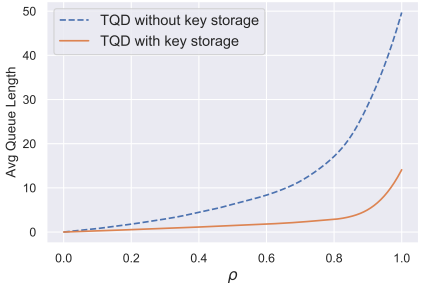


Fig. 6. Delay Performance of the **TQD** policy for broadcast traffic

## VI. Conclusion

In this paper, we designed a secure and provably throughput-optimal routing policy for QKD networks carrying a wide range of traffic. In the future, we plan to extend the policy beyond the trusted node setting.


## VII. Acknowledgement

This work is partially supported by the grant IND-417880 from Qualcomm, USA and a research grant from the Govt. of India for the potential Center-of-Excellence *Intelligent Networks* under the IoE initiative. The computational results reported in this work were performed on the AQUA Cluster at the High Performance Computing Environment of IIT Madras.

## REFERENCES


[1] Abhishek Sinha and Eytan Modiano. Optimal control for generalized network-flow problems. *IEEE/ACM Transactions on Networking*, 26(1):506–519, 2017.

[2] Abhishek Sinha and Eytan Modiano. Throughput-optimal broadcast in wireless networks with point-to-multipoint transmissions. *IEEE Transactions on Mobile Computing*, 2019.

[3] Abhishek Sinha and Eytan Modiano. Network utility maximization with heterogeneous traffic flows. In *2018 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pages 1–8. IEEE, 2018.

[4] Hoi-Kwong Lo and Hoi Fung Chau. Unconditional security of quantum key distribution over arbitrarily long distances. *science*, 283(5410):2050–2056, 1999.

[5] Peter W Shor and John Preskill. Simple proof of security of the BB84 quantum key distribution protocol. *Physical review letters*, 85(2):441, 2000.

[6] Rodney Van Meter. *Quantum networking*. John Wiley & Sons, 2014.

[7] Alexander Ling, Matt Peloso, Ivan Marcikic, Antía Lamas-Linares, and Christian Kurtsiefer. Experimental E91 quantum key distribution. In *Advanced Optical Concepts in Quantum Computing, Memory, and Communication*, volume 6903, page 69030U. International Society for Optics and Photonics, 2008.

[8] Hiroaki Sasaki, Ryutaroh Matsumoto, and Tomohiko Uyematsu. Key rate of the b92 quantum key distribution protocol with finite qubits. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 696–699. IEEE, 2015.

[9] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

[10] Momtchil Peev, Christoph Pacher, Romain Alléaume, Claudio Barreiro, Jan Bouda, W Boxleitner, Thierry Debuisschert, Eleni Diamanti, Mehrdad Dianati, JF Dynes, et al. The secoqc quantum key distribution network in vienna. *New Journal of Physics*, 11(7):075001, 2009.

[11] Xinke Tang, Adrian Wonfor, Rupesh Kumar, Richard V Penty, and Ian H White. Quantum-safe metro network with low-latency reconfigurable quantum key distribution. *Journal of Lightwave Technology*, 36(22):5230–5236, 2018.

[12] Philip G. Evans, Muneer Alshowkan, Duncan Earl, Daniel D. Mulkey, Raymond Newell, Glen Peterson, Claira Safi, Justin L. Tripp, and Nicholas A. Peters. Trusted node QKD at an Electrical Utility. *IEEE Access*, 9:105220–105229, 2021.

[13] Damien Stucki, Matthieu Legre, Francois Buntschu, B Clausen, Nadine Felber, Nicolas Gisin, Luca Henzen, Pascal Junod, Gérald Litzistorf, Patrick Monbaron, et al. Long-term performance of the swissquantum quantum key distribution network in a field environment. *New Journal of Physics*, 13(12):123001, 2011.

[14] Chip Elliott. Building the quantum network. *New Journal of Physics*, 4(1):46, 2002.

[15] Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In *29th IEEE Conference on Decision and Control*, pages 2130–2132. IEEE, 1990.

[16] Michael J Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[17] Saswati Sarkar and Leandros Tassiulas. A framework for routing and congestion control for multicast information flows. *IEEE Transactions on Information Theory*, 48(10):2690–2708, 2002.

[18] Hongyi Zhou, Kefan Lv, Longbo Huang, and Xiongfeng Ma. Security assessment and key management in a quantum network. *arXiv preprint arXiv:1907.08963*, 2019.

[19] Sebastian Nauerth, Florian Moll, Markus Rau, Christian Fuchs, Joachim Horwath, Stefan Frick, and Harald Weinfurter. Air-to-ground quantum communication. *Nature Photonics*, 7(5):382–386, 2013.

[20] Artur K Ekert. Quantum cryptography based on Bell's theorem. *Physical review letters*, 67(6):661, 1991.

[21] Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. An improved lp-based approximation for steiner tree. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 583–592, 2010.

[22] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[23] Michael J Neely. Stability and probability 1 convergence for queueing networks via lyapunov optimization. *Journal of Applied Mathematics*, 2012, 2012.

[24] David Gamarnik. Stability of adaptive and non-adaptive packet routing policies in adversarial queueing networks. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 206–214, 1999.

## APPENDIX A
## PROOF OF THE CONVERSE PART OF THEOREM 1

Consider any admissible arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \boldsymbol{\eta}, \boldsymbol{\gamma})$. By definition, there exists an admissible policy $\pi \in \Pi$ which supports the arrival vector $\boldsymbol{\lambda}$. Without any loss of generality, we may assume the policy $\pi$ to be stationary and the associated DTMC to be ergodic. Let $A_i^{(c)}(0,t)$ denote the number of packets belonging to class $c$ that have arrived at all of their destination(s) along the route $T_i^{(c)} \in \mathcal{T}^{(c)}$ up to time $t$. Recall that each packet is routed along one admissible route only. Thus we can say:

$$\sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) = R^{(c)}(t), \tag{19}$$

where $R^{(c)}(t)$ represents the number of distinct class-$c$ packets received by all destination nodes $\mathcal{D}^{(c)}$ under the action of the policy $\pi$, up to time $t$. We also know that if $A^{(c)}(0,t)$ represents the total number of class-$c$ packet arrivals to the source $s^{(c)}$ up to time $t$, then:

$$A^{(c)}(0,t) \geq \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t), \tag{20}$$

as any packet that has finished its journey along some route $T_i^{(c)} \in \mathcal{T}^{(c)}$ by the time $t$, must have arrived at the source before that. By dividing the inequality (20) by $t$ and taking limit $t \to \infty$, we have:

$$\begin{aligned}\lim_{t\to\infty} \frac{A^{(c)}(0,t)}{t} &\geq \liminf_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) \\ &\overset{(a)}{=} \liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} \\ &\overset{(b)}{=} \lambda^{(c)}.\end{aligned}$$

The equality (a) holds from Eqn. (19) and the equality (b) holds from the definition (1) and the fact that our policy $\pi \in \Pi$ is claimed to securely support the arrival rate $\boldsymbol{\lambda}$. By using SLLN we can say that

$$\lambda^{(c)} = \lim_{t\to\infty} \frac{A^c(0,t)}{t}.$$

From this we can conclude that w.p. 1

$$\lim_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) = \lambda^{(c)}, \quad \forall c \in \mathcal{C} \tag{21}$$

Using the fact that the policy $\pi$ is stationary and the associated DTMC is ergodic, we conclude that the time-average limits exist and they are constant a.s. For all $T_i^{(c)} \in \mathcal{T}^{(c)}$ and $c \in \mathcal{C}$, define

$$\lambda_i^{(c)} \overset{\text{(def)}}{=} \lim_{t\to\infty} \frac{A_i^{(c)}(0,t)}{t}. \tag{22}$$

Using Eqns. (21) and (22) we get

$$\lambda^{(c)} = \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)}. \tag{23}$$

The previous equation (23) proves Eqn. (1) that there exists a non-negative flow decomposition of the incoming packets amongst the admissible routes.

For the second part of proof, we consider an edge $e \in E$ in the graph $\mathcal{G}$. Let $A_e(0,t)$ be the number of packets that have crossed edge $e$ till time $t$ under the action of the policy $\pi$. We have that:

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \;\leq\; A_e(0,t) \overset{(a)}{\leq} \sum_{\tau=0}^{t} K_e(\tau), \tag{24}$$

and

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \;\leq\; A_e(0,t) \overset{(b)}{\leq} \sum_{\tau=0}^{t} \gamma_e, \tag{25}$$

where the left-most sides of the inequalities (24) and (25) denote the number of delivered packets which has crossed the edge $e$ by the time $t$. This is clearly a lower-bound on $A_e(0,t)$. The inequality (a) in Eqn. (24) arises from the fact that the total number of quantum keys generated by the QKD link $e$ up to time $t$ is an upper bound to the number of packets that have crossed the edge till time $t$. Similarly, the inequality (b) in Eqn. (25) arises from the fact that the number of packets that have crossed edge $e$ till time $t$ cannot be greater than the cumulative capacity of the link up to time $t$.

Combining inequalities (24) and (25), we have a tighter bound:

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \;\leq \min\left(\sum_{\tau=0}^{t} K_e(\tau), \sum_{\tau=0}^{t} \gamma_e\right). \tag{26}$$

Dividing both sides of the above inequality by $t$ and taking the limit $t \to \infty$ we get

$$\lim_{t\to\infty} \sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \frac{A_i^{(c)}(0,t)}{t} \leq \min\left(\lim_{t\to\infty} \frac{\sum_{\tau=0}^{t} K_e(\tau)}{t}, \lim_{t\to\infty} \frac{\gamma_e t}{t}\right) \tag{27}$$

Using Eqn. (22) on LHS and SLLN on first term in the RHS of Eqn. (27), we get

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \min(\eta_e, \gamma_e) = \omega_e \tag{28}$$

By the definition in Eqn. (2) we see that the condition that no edge is overloaded translates to

$$\lambda_e \leq \omega_e \tag{29}$$

This establishes the converse part of Theorem 1. ■

## APPENDIX B
## THROUGHPUT-OPTIMALITY OF TQD

For any class $c \in \mathcal{C}$, let $A^{(c)}(0,t)$ be the total number of incoming packets belonging to class $c$ up to time $t$. The total number of packets $R^{(c)}(t)$ commonly received by all destination nodes $\mathcal{D}^{(c)}$ of class $c$ can be bounded as follows:

$$A^{(c)}(0,t) - \sum_{e \in E} X_e(t) - \sum_{e \in E} Y_e(t) \overset{(a)}{\leq} R^{(c)}(t) \overset{(b)}{\leq} A^{(c)}(0,t). \tag{30}$$

Here the first inequality $(a)$ arises from the observation that if a packet $p$ of class $c$ has not reached all destination nodes $\mathcal{D}^{(c)}$, then at least one copy of it must be present in some of the physical queues. Inequality $(b)$ states the obvious fact that the number of packets received till time $t$ is less than the number of packets that have arrived at the source till time $t$. Since the TQD policy is proven to be rate stable, we know that

$$\lim_{t \to \infty} \frac{\sum_{e \in E} X_e(t)}{t} = 0 \quad \text{and} \quad \lim_{t \to \infty} \frac{\sum_{e \in E} Y_e(t)}{t} = 0.$$

Thus, dividing both sides of the inequality (30) by $t$ and taking the limit $t \to \infty$, we get

$$\lim_{t \to \infty} \frac{A^{(c)}(0,t)}{t} \leq \lim_{t \to \infty} \frac{R^{(c)}(t)}{t} \leq \lim_{t \to \infty} \frac{A^{(c)}(0,t)}{t}.$$

Thus:

$$\lim_{t \to \infty} \frac{R^{(c)}(t)}{t} = \lim_{t \to \infty} \frac{A^{(c)}(0,t)}{t} = \lambda^{(c)}, \forall c \in \mathcal{C}.$$

This shows that the TQD policy is secure and throughput optimal. ■