Enhancing Key Availability and Efficiency in QKDNs via Decentralized Key Flooding

María Álvarez Roa
Department of Electrical Engineering
Eindhoven University of Technology
Eindhoven, The Netherlands
m.alvarez.roa@tue.nl

Sebastian Verschoor
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
s.r.verschoor@uva.nl

Simon Rommel
Department of Electrical Engineering
Eindhoven University of Technology
Eindhoven, The Netherlands
s.rommel@tue.nl

***Abstract*—The key flooding protocol in Quantum Key Distribution Networks (QKDNs) is a fully decentralized strategy that proactively distributes key material to pre-establish end-to-end key pools, enabling immediate delivery to clients. We enhance this protocol to increase key availability and overall distribution efficiency. The protocol has been implemented in an in-house developed, fully functional Key Management System (KMS), and the evaluation has been carried out over a Docker-based simulated network with a representative topology, demonstrating practicality and feasibility in realistic deployment scenarios. Experimental results show that, under moderate network load, the enhanced protocol significantly improves key request success rates by up to 5.6 % while authentication overhead is reduced by up to 59 % compared to the baseline flooding approach. The protocol maintains stable performance under stress conditions, preserving robustness against increased traffic.**

***Index Terms*—QKD, QKD networks, key relaying, KMS**

I. Introduction

Quantum Key Distribution (QKD) is an emerging technology that addresses the security challenges associated with future computational advances, particularly those introduced by quantum computing [1]. Unlike classical cryptographic, which rely on the computational hardness of problems that can be efficiently solved by known quantum algorithms [2], QKD derives its security from the laws of quantum mechanics rather than assumptions about an adversary's computational power. Assuming a pre-shared key, it allows two parties to exchange cryptographic keys securely, providing information-theoretic security (ITS) [3]. Once the keys have been securely distributed, they can be used for arbitrary cryptographic purposes, including the one-time pad (OTP) for ITS encryption or the Advanced Encryption Standard (AES) for quantum-safe encryption [4], [5].

QKD is inherently a point-to-point technology with distance limitations. These distance restrictions, combined with low key generation rates present the main challenges in the deployment of QKD networks. The most widely adopted practical solution to overcome the rate-distance constraint [6] is to build QKD networks using trusted relay nodes. While trusted repeaters do introduce more trust assumptions for security, until quantum repeaters become viable [7], trusted relays offer a feasible means of extending network where keys are stored in QKD nodes and relayed to other QKD nodes not directly connected to the source via a QKD link [8].

These networks enable secure key exchange between users across arbitrary distances through a process known as key relaying, coordinated by a key management system (KMS). In this approach, secret key material generated by QKD modules and subsequently managed by the KMS are consumed along the relay path to establish keys between the remote users. Thus, key material can be used to serve key requests from cryptographic applications and during the relay process for OTP encryption and authentication of specific messages [9]. Because the secret key rate available for generating new quantum keys is inherently limited, the use of key material must be carefully managed. The efficiency of the relay process therefore becomes crucial as inefficient key consumption can lead to the depletion of available quantum keys in the key pools. Thus, effective key management strategies are needed to maximize the use of the generated keys, ensuring that relay nodes retain sufficient reserves for relaying processes while still keeping enough key material readily available to handle key request from clients.

Within a QKD network (QKDN), the KMS plays a central role in coordinating key storage, relaying, and allocation to clients. Various strategies can be used to manage how relay nodes distribute and establish key material between end nodes, as well as how that material is consumed during the process. While standardization bodies such as ITU-T and ETSI envision centralized SDN-based control, QKDN design is still at an early stage, and it remains unclear whether such approaches will be universally adopted. As in classical networking, the decision between centralized and decentralized solutions will likely depend on topology and application requirements. In [10], key flooding is presented, a new protocol for the management, storage, and distribution of key material in a QKDN. It proactively distributes quantum keys from adjacent to non-adjacent key pools across the network in a decentralized, flooding-like, manner. By pre-distributing key material between any pair of endpoints, this approach aims that end-to-end keys are immediately available for client requests, reducing key delivery delay. In this paper, we build upon our previous work [11] and propose enhancements for the key flooding protocol, evaluated within the framework of a QKDN

Quantum Delta NL and EU's Digital Europe Programme (QCINed).

978-3-903176-78-2

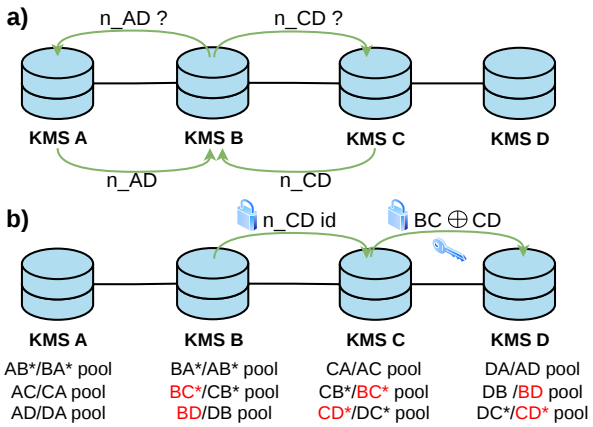


Fig. 1: Key distribution scheme for key flooding protocol. OTP encryption, denoted as $\oplus$, is used to protect client keys and the padlock indicates ITS message authentication.

with the aim of comparing their performance in terms of key request success rate and authentication overhead resulting from relaying. Furthermore, the protocol is implemented in an in-house developed fully functional KMS, and the evaluation has been carried out over a Docker-based simulated network.

Our results show that under moderate demand, the enhanced key flooding protocol can increase the overall key request success rate by up to 5.6 % and reduces authentication overhead up to 59 %. In scenarios with high demand, the overall gains are smaller due to depletion of available key material; however, lower-performance links still experience measurable benefits, demonstrating enhanced key availability even when the network is under stress. These findings demonstrate that the proposed enhancements not only improve resource usage in a QKDN but also strengthen the robustness and reliability of key distribution across varying network conditions.

II. Key flooding protocol enhancements

In most deployed QKD networks, extending the distance between endpoints is achieved through trusted relay nodes [6], but this approach relies on consuming additional key material. Because the secret key rate available for generating new quantum keys is inherently limited, the use of key material must be carefully managed. The efficiency of the relay process therefore becomes crucial for large-scale quantum-secured networks as inefficient key consumption can lead to the depletion of available quantum keys. Thus, effective key management strategies are needed to maximize the use of the generated keys, ensuring that relay nodes retain sufficient reserves for relaying processes while still keeping enough key material readily available to handle key request from clients.

The key flooding proposal presented in [11], considered as an alternative to the well-known on-demand hop-by-hop relaying [9], is designed to reduce delays in key delivery, which is particularly beneficial for applications with strict latency constraints while also simplifying routing in quantum-secured networks. Furthermore, the absence of a central controller both simplifies implementation and eliminates a potential single point of failure. The protocol distributes key material from adjacent to non-adjacent key pool enhancements anywhere in the network ahead of time in a decentralized manner. Key pools between any pair of endpoints are established, allowing keys to be ready for delivering when they are requested by clients, with all possible pools being populated as part of the protocol's design. This advance distribution significantly reduces key delivery delays.

To fill all key pools in the network, each node periodically identifies non-adjacent pools requiring keys, i.e., pools where the amount of key material is below the minimum target number of keys for immediate client use, referred to as client key reserve. Once the QKD node identifies which end-to-end key pools need replenishment, pools with a lower number of hops between nodes will be selected to be filled first. Once such a target pool is chosen, the initiator node starts the relaying process, selecting an intermediate neighbor to forward the keys and determining the appropriate relay block size based on available resources. Fig. 1 shows the process used to populate non-adjacent key pools within the QKDN and considered here as the baseline [11]. Key pools between adjacent nodes are denoted by an asterisk (*) in the pool name. In the example shown, the target pool to be filled is BD, with KMS B acting as the initiator and KMS D as the destination.

In the first stage of the process (Fig. 1 a)), KMS B queries each neighboring KMS to determine the number of available keys that can be used to reach the target node D. It then selects the neighbor offering the shortest path in terms of hop count, KMS C in the example. The number of keys to be relayed is set to the minimum of the reported available keys toward D, denoted n_CD, and the number of keys available for relaying in the corresponding adjacent key pool, BC*. In the example, we assume the number of available keys reported, n_CD, is the minimum. In the second stage (Fig. 1 b)), KMS B transfers n_CD keys from the adjacent key pool BC* into the non-adjacent pool BD. The relaying KMS, C, receives from KMS B the identifiers of these n keys, which specify the key material to be forwarded to KMS D. The selected keys are extracted from pool BC* and relayed to D, where they are stored in pool BD. An ITS message authentication code, that consumes a QKD key, is computed over the entire message marked with a padlock. In this approach, only the message carrying the actual key material is additionally protected using OTP encryption, which also consumes QKD keys. In this example, that message is sent from KMS C to KMS D where keys used for encryption and authentication will be taken from the key pool CD*.

In terms of key consumption, the baseline scenario incorporates an adjustment to the protocol proposed in [11]. Specifically, an authentication key is now consumed each time key material is delivered to the clients, and is used to authenticate the inter-KMS notification message indicating which keys are consumed, thereby ensuring proper synchronization.

We further propose enhancements, which aim to improve

TABLE I: Summary of main characteristics of baseline and enhanced scenarios

Scenario	Main Characteristics
Baseline	Relay bytes are evenly divided among all neighbors based on node degree. Adjacent node selection relies on hop distance only and ignores source–neighbor key availability.
E1a	Relay bytes are divided among neighbors excluding the initiator node.
E1b	Relay byte allocation is dynamically recalculated; active neighbors that use the pool to fill their own end-to-end pools receive more bytes.
E1c	Extend E1b by reserving a portion of key material for the node's own end-to-end pools to ensure internal availability.
E2	Candidate end-to-end pools are selected based on minimum hop distance; ties are resolved by lower pool levels.
E3	Neighbor selection considers hop distance and key availability; ties are resolved using the minimum available keys between source–neighbor and neighbor–target.

efficiency of the QKDN by improving how resources, i.e., key material generated by QKD modules, are allocated and how candidate end-to-end pools and neighbors nodes are prioritized. Primary features of the baseline protocol (BL) and proposed enhancements (E1–E3) are summarized as follows:

BL Baseline: The number of available relay bytes in each neighbor node (n), derived from the pool between neighbor and destination node, is computed by dividing total key material reserved for authenticating relay messages and OTP encryption of client keys during relaying, by the number of neighbor nodes (known as node degree). Non-adjacent key pools are filled based on hop distance between end nodes, with ties resolved by random selection. The adjacent node (a) selected to perform the flooding is chosen among neighboring nodes with available key material. If multiple neighbors have available keys, the node with the lowest hop distance to target is selected. If multiple neighboring nodes have available key material and same hop distance, the first eligible node in the list is selected. The baseline does not consider source-neighbor key availability for the selection of the adjacent node.

E1 Relay Byte Allocation:

E1a: Once a pool is selected to be filled, n is calculated by dividing the link bytes available in the pool by the node degree excluding the initiator node. The calculation avoids making an unncessary reservation of keys for the source of the request, thereby increasing the amount of key material distributed across the network.

E1b: Independently of the adjustment introduced in E1a, n is recalculated is performed whenever new blocks are stored in a key pool. To further improve resource utilization, we differentiate between active and non-active neighbors by assigning more relay bytes to active nodes, i.e., nodes that use this key pool to fill their own end-to-end pools.

E1c: Extends E1b by reserving additional bytes specifically for filling the node's own end-to-end pools. By setting aside a dedicated portion of key material, the node prioritizes that its internal pool requirements are met even under high relay demand from neighbors.

E2 End-to-End Pool Selection: Candidate end-to-end pools are selected by hop distance between source and destination nodes. In cases where multiple pools have equal hop distances, ties are resolved by comparing pool levels, giving priority to the pool with the lower level.

E3 Neighbor Node Selection: The selection of a is done using hop distance to target and available keys. Among neighbors with available keys, the node with the minimum hop distance to the target is selected. If multiple neighbors exhibit the same minimum hop distance, a secondary decision metric is applied, defined as the minimum between neighbor–target and source–neighbor available keys.

Table I summarizes the main characteristics of the baseline and proposed enhancement scenarios. The baseline represents the reference design. The proposed enhancements are integrated into a functional KMS and evaluated to assess their impact on network performance. Each enhancement was simulated separately to isolate its individual effect, as well as in combination with the others to evaluate their cumulative impact on the system. The simulation framework employed for this evaluation is presented in the following section.

III. Simulation Framework and Scenarios

Docker containers are used to model and deploy the different components of the QKDN, with each network entity running as an isolated container, ensuring modularity and scalability. The Docker-based setup deploys an in-house developed fully functional KMS in multiple instances inside containers, enabling real communication between all KMS modules. The experimental setup provides a high-fidelity reproduction of operational KMS behavior and system interactions. The QKDN architecture [12] is modeled across its three principal layers: the quantum layer, the key management layer, and the application layer. Each layer is represented as follows.

Each QKD device is modeled as an application running within its own container. Key material is generated locally using a pseudo-random process and is periodically uploaded to the associated KMS. To closely model the performance of a QKDN, secret key rate (SKR) is estimated from the decoy-state BB84 protocol. The SKR is computed as a function of

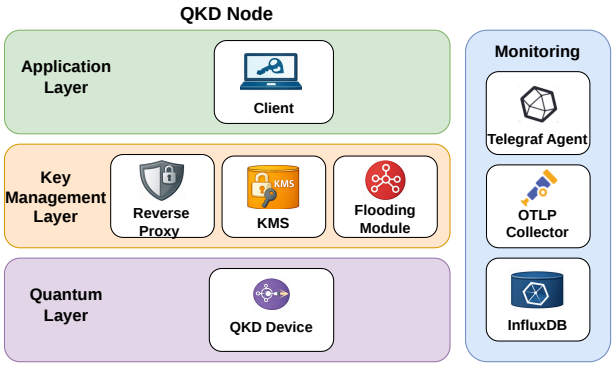


Fig. 2: Docker containers deployed per QKD node, illustrating the three-layer QKDN architecture. Each node hosts a client, a QKD device, and key management components, comprising a KMS, a flooding module, and a reverse proxy, while monitoring is enabled through OTLP, Telegraf, and InfluxDB.

link distance, using the parameter set and analytical model reported in [8], assuming a quantum channel operating at a wavelength of 1550 nm. This approach ensures that key generation dynamics in the simulator reflect realistic physical-layer constraints observed in practical QKD systems.

Each node also hosts a client container, which asynchronously requests keys from the KMS. The arrival times of key requests are modeled with an exponential distribution, while the destination QKD node for each request is chosen randomly according to a uniform distribution. Each KMS runs in a dedicated container, extended with a flooding module deployed in a separate container, which communicates with the KMS through one of its communication endpoints. A reverse proxy container, responsible for mutual authentication between the KMS and connecting entities, is deployed to make the simulation closely matched to a real-world deployment.

The in-house KMS also offers monitoring capabilities by exporting metrics and traces through the OpenTelemetry Protocol (OTLP). It uses an OTLP exporter to send telemetry data to the corresponding Telegraf agent, where the OTLP plugin processes the data before it is stored in InfluxDB. Fig. 2 illustrates the deployment of Docker containers for each QKD node, capturing the three-layer QKDN architecture. The figure also highlights the monitoring setup required.

To model the QKD network and evaluate the flooding protocol and the novel proposed enhancements, the NSFNET topology [13], consisting of 14 nodes and 22 links, with average node degree of 3.14 and maximum hop distance of 3, is used. We analyze network performance under varying client key consumption rates. We consider a moderate-demand scenario in which key request size follows a uniform distribution between 192 bytes and 640 bytes, corresponding to an average request size of 416 bytes, and a high-demand scenario with an increased range of 320 bytes to 960 bytes, resulting in an average request size of 640 bytes. The timing of key request from clients follows an exponential distribution with a 2 s average interval. This means that, on average, each node receives one request every two seconds while exact arrival times vary, representing random nature of request arrivals.

TABLE II: Simulation parameters

Parameter	Values
Average inter-request interval	2 s
Key size	32 byte
Inter-arrival time of key blocks	40 s
Average bytes requested	416; 640 byte
Max Capacity	50 kbyte
Client key reserve	1500 byte
Flooding trigger interval	20 s

Table II summarizes the key system parameters affecting performance. For simplicity, all key pools are assumed to have the same maximum capacity. In the key flooding strategy, we vary the flooding trigger interval, which controls how often non-adjacent key pools are refilled. This process is triggered asynchronously across nodes, with the same frequency for each one. The client key reserve refers to the minimum number of client keys allocated to each key pool. Once this minimum target is achieved, surplus keys can be used for OTP encryption and authentication for relaying. Note that a fixed amount of 32 bytes per message is consumed for authentication.

To improve accuracy and reduce random effects, multiple two-hour simulation runs were performed per scenario. Starting from empty pools, the system stabilizes after ∼30 min, so this initial period is excluded from the metrics presented to avoid distorted results from the fact that all key requests fail due to a shortage of key material during the initialization of the QKD network. The data presented shows the average of results, with a standard deviation maintained below 1 % to ensure minimal variability.

IV. Experimental Results

The success rate is a key performance metric representing the proportion of key requests that are successfully fulfilled. A key request is considered successful when a client key is available for delivery in response to the request. Fig. 3 shows the average value of the overall success rate across the entire network for the BL scenario and the proposed enhancements under both moderate and high network load conditions. In addition, the standard deviation is indicated with error bars. The results show that the combined enhancements E1a-2-3 yield the highest improvement, with a maximum increase of 5.6 % in overall success rate under moderate key demand and 0.9 % under high key demand. E1b-2-3 and E1c-2-3 achieve gains of 5.1 % and 4 %, respectively, under moderate demand, but shows no improvement when the network is under stress. These results indicate that while some combinations of enhancements substantially improve key delivery under typical network load, their effectiveness decreases under high traffic conditions. At the same time, E1a-2-3 demonstrates higher robustness, consistently maintaining improved success rates across both moderate and high demand scenarios, although the gains under high demand are less pronounced.

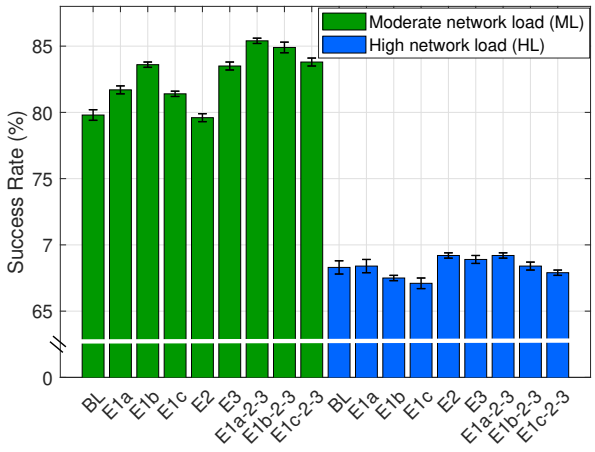


Fig. 3: Comparison of the overall success rate for moderate (green) and high (blue) network load.

TABLE III: Success rate improvement under moderate network load (ML) compared to the baseline

Scenario	Included Enhancements	Success Rate Improvement (ML)
BL	—	-
E1a	E1a	1.9%
E1b	E1b	3.8%
E1c	E1c	1.6%
E2	E2	-
E3	E3	3.7%
E1a-2-3	E1a + E2 + E3	5.6%
E1b-2-3	E1b + E2 + E3	5.1%
E1c-2-3	E1c + E2 + E3	4%

Table III details the success rate improvements obtained under moderate network load (ML) for each individual enhancement and for their combined configurations. Among the standalone enhancements, E1b delivers the highest performance gain, increasing the success rate by 3.8 %, closely followed by E3 with an improvement of 3.7 %. When enhancements are combined, their impact becomes more pronounced. All composite scenarios outperform the individual configurations. In particular, E1a-2-3 achieves the largest improvement under moderate load, as already discussed in the analysis of Fig. 3. Note that overlapping effects between enhancements prevent their gains from being strictly additive, resulting in a combined improvement that is lower than the sum of individual contributions. In addition, we observe that E2 provides improvements when combined with other enhancements, whereas it shows no impact on the success rate when evaluated independently. Results for the combinations with and without E2 are not shown here due to space limitations.

Fig. 4 shows the average success rate per link for BL (a, c), E1a-2-3 (b, c) and E1b-2-3 (c, f), with the first and second rows corresponding to moderate and high network load, respectively. While E1b-2-3 does not raise the overall network success rate when high demand is considered, it noticeably improves the success rate of links that exhibit lower success rates in the BL scenario. This improvement comes at the expense of a slight reduction in the success rate of links that were already performing well, effectively redistributing key delivery performance to achieve a more balanced network. A similar behavior is observed for E1c-2-3 (not shown), highlighting that certain enhancements can mitigate weak links even if they do not significantly increase the overall average. Although E1a-2-3 produces the largest overall increase in the network-wide success rate, a per-link analysis reveals additional insights. In particular, the number of links with a success rate below 15 % is reduced under both moderate and high network load when the combination of enhancements E1b-2-3 is applied. This indicates that while E1a-2-3 maximizes the overall success rate, E1b-2-3 more effectively strengthens weaker links, improving the performance of underperforming connections even if the overall network average is less impacted.

TABLE IV: Average relay block size compared to the baseline for different scenarios and hop distances.

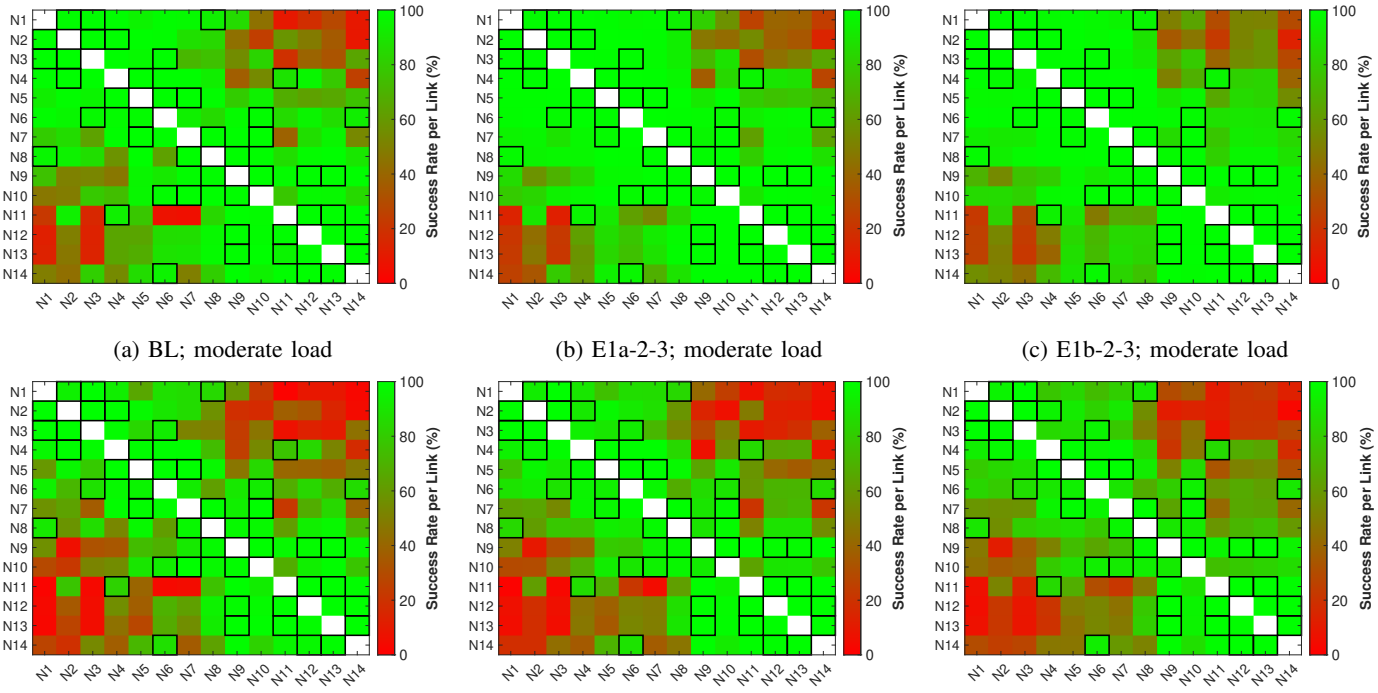
	Avg. Relay Block Size (bytes)	
Scenario	**Hop distance = 2**	**Hop distance = 3**
BL ML	3163 ± 14	740 ± 19
E1a-2-3 ML	3348 ± 12	930 ± 5
E1b-2-3 ML	3932 ± 96	1807 ± 68
E1c-2-3 ML	2849 ± 60	1485 ± 17
BL HL	3071 ± 78	737 ± 13
E1a-2-3 HL	2719 ± 53	728 ± 13
E1b-2-3 HL	3505 ± 39	1398 ± 28
E1c-2-3 HL	2495 ± 38	1288 ± 5

Table IV shows the average relay block size allocated to fill end-to-end pools between nodes at varying hop distances. Within the NSFNET topology, the maximum separation between any two nodes is three hops. During flooding, each authenticated message consumes one quantum key, so smaller key blocks result in higher relative authentication overhead. Using larger blocks reduces overhead per transferred bit, improving key distribution efficiency. From the table, we can see that E1a-2-3 slightly increases the average block sizes compared to BL for moderate network load, resulting in a modest overhead reduction, while E1b-2-3 significantly enlarges the blocks, achieving authentication overhead reductions of 19.6 % and 12.4 % for two-hop links and 59 % and 47.3 % for three-hop links under moderate and high load, respectively.

V. Conclusions

In this work, we proposed and evaluated several enhancements to the key flooding protocol for QKDNs. Building on the original protocol, which proactively distributes key material from adjacent to non-adjacent pools in a decentralized manner, the enhancements focus on improving key allocation. Our evaluation, conducted in a Docker-based simulated network with a fully functional KMS implementation, demonstrates the feasibility of these enhancements in a realistic environment.

Under moderate network demand, the enhanced protocol achieves up to a 5.6 % increase in overall key request success



(a) BL; moderate load (b) E1a-2-3; moderate load (c) E1b-2-3; moderate load

(d) BL; high load (e) E1a-2-3; high load (f) E1B-2-3; high load

Fig. 4: Average success rates per link for different scenarios. Note that key pools between adjacent nodes, are marked with black squares and blank results indicate no key pool exist.

rate, while distributing larger key blocks reduces authentication overhead by up to 59 %. Although overall gains are smaller under high demand due to network saturation, per-link analysis shows that low-performance links benefit from improved key availability, demonstrating that the enhancements can strengthen weaker parts of the network even under stress. These results highlight that the enhancements not only improve key material usage but also improve the robustness and reliability of key distribution across varying load conditions.

Acknowledgment

This work was partly supported by the Dutch Ministry of Economic Affairs and Climate Policy (EZK) through Quantum Delta NL and by the EU's Digital Europe Programme (QCINed, grant no. 101091656).

References

[1] D. J. Bernstein and T. Lange, "Post-quantum cryptography," *Nature*, vol. 549, no. 7671, Sep. 2017.

[2] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96. New York, NY, USA: Association for Computing Machinery, 1996, p. 212–219. [Online]. Available: https://doi.org/10.1145/237814.237866

[3] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," *Theoretical Computer Science*, vol. 560, p. 7–11, Dec. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.tcs.2014.05.025

[4] C. Rubio García, S. Rommel, S. Takarabt, J. J. Vegas Olmos, S. Guilley, P. Nguyen, and I. Tafur Monroy, "Quantum-resistant transport layer security," *Computer Communications*, vol. 213, pp. 345–358, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366423004012

[5] C. Rubio García, A. Cano Aguilera, C. Stan, J. José Vegas Olmos, S. Rommel, and I. Tafur Monroy, "Enhanced network security protocols for the quantum era: Combining classical and post-quantum cryptography, and quantum key distribution," *IEEE Journal on Selected Areas in Communications*, vol. 43, no. 8, pp. 2765–2781, 2025.

[6] 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.

[7] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, "Quantum repeaters: The role of imperfect local operations in quantum communication," *Phys. Rev. Lett.*, vol. 81, pp. 5932–5935, Dec 1998. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.81.5932

[8] C. Stan, D. Verchere, J. J. V. Olmos, I. T. Monroy, and S. Rommel, "Dynamic-threshold-based pre-relaying for enhanced key allocation in quantum-secured networks," *J. Opt. Commun. Netw.*, vol. 17, no. 3, pp. 233–248, Mar 2025. [Online]. Available: https://opg.optica.org/jocn/abstract.cfm?URI=jocn-17-3-233

[9] M. Peev, C. Pacher, R. Alléaume, and others., "The SECOQC quantum key distribution network in Vienna," *New Journal of Physics*, vol. 11, no. 7, p. 075001, jul 2009. [Online]. Available: https://dx.doi.org/10.1088/1367-2630/11/7/075001

[10] M. A. Roa, S. Verschoor, and S. Rommel, "Efficiency analysis of two key relaying architectures for qkd networks," in *ECOC 2024; 50th European Conference on Optical Communication*, 2024, pp. 1389–1392.

[11] M. Alvarez Roa, C. Stan, S. Verschoor, I. Tafur Monroy, and S. Rommel, "Decentralized key distribution versus on-demand relaying for qkd networks," *Journal of Optical Communications and Networking*, vol. 17, no. 8, pp. 732–742, 2025.

[12] ITU-T, "Quantum key distribution networks – Key management," International Telecommunication Union, Standard Y.3803, Dec. 2020. [Online]. Available: https://www.itu.int/rec/T-REC-Y.3803-202012-I/en

[13] X. Chen, B. Li, R. Proietti, H. Lu, Z. Zhu, and S. J. B. Yoo, "Deep-RMSA: A deep reinforcement learning framework for routing, modulation and spectrum assignment in elastic optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4155–4163, 2019.