A Distributed Protocol for Multipath Key Reinforcement in Virtual Private Networks

David Schatz
Technische Universität Ilmenau
david.schatz@tu-ilmenau.de

Hedwig Koerfgen
Universität der Bundeswehr München
hedwig.koerfgen@unibw.de

Guenter Schaefer
Technische Universität Ilmenau
guenter.schaefer@tu-ilmenau.de

***Abstract*—Due to the quantum threat, hybridization and agility in cryptography is more import than ever. For key exchange, the consensus is an immediate transition to post-quantum cryptography (PQC), in the form of key encapsulation mechanisms (KEMs), in combination with the traditional Diffie-Hellman key exchange. However, hybridization does not need to be limited to classical and post-quantum asymmetric cryptography. In this article, we study multipath key reinforcement (MKR) as an additional hybridization for quantum-resistant virtual private networks (VPNs): Security established on some links can be propagated through a network to reinforce additional connections. To the best of our knowledge, this article proposes the first distributed protocol to perform MKR in VPNs. Our simulation-based evaluation shows that our protocol is able to increase security within the VPN quickly and with low overhead, even under a worst-case initial compromise. These results demonstrate that MKR is a very cost-effective building block for hybrid, quantum-resistant VPN.**

***Index Terms*—Multipath Key Reinforcement, Quantum Resistance, Virtual Private Networks, Hybrid Key Exchange, Path Hopping**

I. Introduction

The security of classical asymmetric cryptography such as RSA, ElGamal, and Diffie-Hellman is threatened by quantum computers, because a sufficiently large quantum computer could solve the prime factorization and the discrete logarithm problems efficiently [1], [2]. For key exchange, this motivates an immediate transition to post-quantum cryptography (PQC) due to "store now, decrypt later" attacks. This is reflected in the recent standardization of the first PQC key encapsulation mechanism (KEM), namely ML-KEM [3]. However, trust in PQC algorithms is not quite as high as in their classical counterparts, yet. One reason for this is that their actual integration into security protocols and the resulting implementation security is not studied as well, yet. Accordingly, many security agencies currently recommend solutions that focus on *hybridization* and *cryptographic agility* [4], [5].

For key exchange, these recommendations suggest a combination of PQC KEMs with the traditional Diffie-Hellman key exchange in an agile way. However, additional measures are also discussed: *Quantum key distribution (QKD)* offers security based on the laws of quantum physics, but requires specialized infrastructure and faces implementation-security challenges [6]. *Pre-shared keys (PSKs)* are considered highly secure, but manual distribution does not scale to large networks and opportunistic approaches [7] cannot guarantee the distribution of PSKs for all pairs of endpoints. *Multipath key reinforcement (MKR)* distributes key materials over multiple paths and combines them at endpoints. This is especially appealing because it allows security established on some links (e.g., via PQC, QKD, PSKs) to be propagated to further links in a network, as depicted in Fig. 1. However, recent proposals mainly focus on theoretical properties and limits of MKR. To the best of our knowledge, prior work does not propose how to route MKR secrets in overlay networks, nor a robust synchronization mechanism under packet loss. Prior work also lacks a quantitative evaluation of how *fast* MKR is able to secure an overlay network in practice and at what cost.

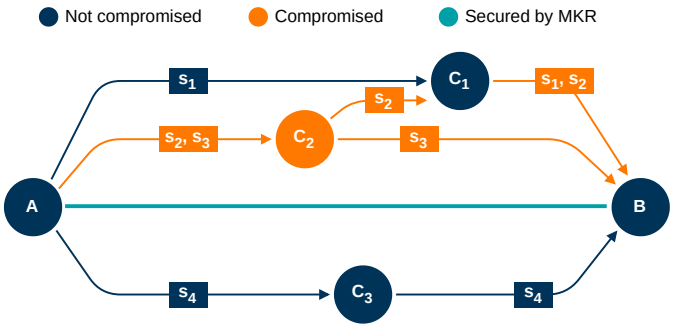

Fig. 1. Example for MKR between nodes A and B, sending four secrets $s_i, 1 \leq i \leq 4$ via four different paths. While three of the four paths are assumed to be compromised in this example, the path taken by s_4 is not compromised. By subsequently using a resilient combination of all $s_i, 1 \leq i \leq 4$ as key material to reinforce the direct connection between A and B, the security of path (A, C_3, B) propagates to (A, B).

To bridge this gap, we present a simple randomized and distributed routing algorithm, together with a simple synchronization protocol. As example scenario, we use virtual private network (VPN) infrastructures. Still, our approach is equally applicable to any network infrastructure that can be modelled as an *overlay network* with end-to-end authentication capabilities. Using a simulation-based approach, we answer two main research questions:

1) How fast can MKR propagate security to all overlay links for which initially, at least one secure path exists?
2) How much overhead does MKR introduce?

This work is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr [project MuQuaNet]. dtec.bw is funded by the European Union - NextGenerationEU.

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

The remaining article is structured as follows: In Sec. II, we define the assumed communication and threat model, followed by requirements for an MKR protocol based on that model in Sec. III. Related work is discussed in Sec. IV. We present our approach in Sec. V and quantitatively evaluate it in Sec. VI. Sec. VII concludes our article.

II. Communication and Threat Model

Our communication model is inspired by (IPsec) *VPN overlay networks*. That is, VPN connections form an overlay on top of the untrusted transport network. We model the VPN as an undirected graph $G = (V, E)$, with endpoints V and VPN connections $E \subseteq \binom{V}{2}$. The terms (VPN) connection, link, and edge are used interchangeably throughout this article. Nodes adjacent to a node $v \in V$ are called its *neighbors*, denoted $N(v)$.

Connections are established by a hybrid *authenticated key exchange*, e.g., as realized by the IKEv2 protocol (Internet Key Exchange, protocol version 2) [8]: A hybrid key exchange integrates classical (e.g., Diffie-Hellman) and post-quantum mechanisms (e.g., ML-KEM [3]). Hybrid authentication could be based on hybrid signatures, e.g., ECDSA combined with SLH-DSA [9]. The shared session key is subsequently used to encrypt all messages on the connection.

We further assume an extended Dolev-Yao attacker [10]:

- Attackers may observe, modify, drop, duplicate and inject new messages on every connection.
- They may break classical asymmetric schemes and a subset of post-quantum schemes, resulting in *compromised connections*.
- However, we assume they cannot forge hash-based digital signatures within seconds.
- Some *nodes may also be compromised*, leaking all session keys of adjacent links.

III. Requirements

The following requirements for an MKR protocol address confidentiality, integrity, and availability in the context of the defined communication and threat model, categorized into functional, security, and non-functional requirements.

A. Functional Requirements

The functional requirements for MKR are:

F1) *Periodic Execution:* MKR must be performed periodically for each connection $e \in E$, as a single execution cannot guarantee that an uncompromised path is used. This also implies that as many MKR key materials as possible must be combined over time to reinforce connection e (across all executions, and across all paths per execution). The number k of paths per execution and the rate λ of executions (or analogously, the MKR interval $1/\lambda$) should be configurable.

F2) *Synchronized Selection of Key Material:* For each connection $e = \{u, v\}$ and every point in time, both endpoints u, v must maintain a synchronized view of a subset of MKR key materials available on both sides. Such a synchronized subset is consequently used to reinforce the connection e. This requirement must hold despite asynchronous delivery and potential packet loss.

B. Security Requirements

MKR must be implemented securely as follows:

S1) *Resilient Combination of Key Material:* Related to requirement F1, the combination of many MKR key materials over time must be *resilient* to partial compromise. Especially, the resulting key should not be reconstructable by attackers even if they know *all but one input* to the combination function.

S2) *Secure Generation of Key Material:* Individual key materials exchanged via MKR must be generated by a pseudo-random number generator suited for cryptographic use.

S3) *Authenticated Selection of Key Material:* As attackers may temporarily impersonate honest endpoints of compromised connections $e = \{u, v\}$, the selection process should be freshly authenticated. Otherwise, attackers might be able to "desynchronize" u and v, rendering the combined key material useless.

S4) *Availability:* The protocol must tolerate message loss and active interference by an attacker, ensuring that attackers cannot prevent the synchronization of uncompromised MKR key material by selectively dropping synchronization messages.

S5) *Graceful Degradation:* Compromise of individual nodes must not degrade the security and availability of unrelated connections.

C. Non-functional Requirements

From a non-functional perspective, the following requirements are identified:

N1) *Bandwidth overhead* introduced by MKR should be low.

N2) *Computational overhead* at each node should be low.

N3) *Scalability:* The MKR protocol should scale to very large overlay networks with thousands of nodes.

IV. Related Work

MKR is closely related to the *Perfectly Secure Message Transmission (PSMT)* problem [11], studying secure communication in a partially compromised network. While PSMT provides strong theoretical guarantees, it leaves open questions regarding complexity and practical deployments [12]. Early MKR approaches emerged from key establishment in Wireless Sensor Networks (WSNs), where combining secret sharing with multipath routing improves confidentiality, as demonstrated by [13]. However, these results rely on assumptions of largely independent, randomly formed paths, which do not hold in structured networks such as VPN overlays.

An alternative to secret sharing approaches is *path hopping* where paths change over time to reduce exposure. In [14], path hopping is described as a cryptographic primitive with the goal of information-theoretic security against quantum adversaries. Recent work like [15] incorporates realistic constraints (QoS, capacity, path overlap) while requiring SDN-based routing

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

control. Yet, evaluation metrics are mostly traffic-centric and focus on packet arrival, not on confidentiality.

Multipath approaches in VPNs also target availability rather than security [16]–[18]. Some architectures combine multiple cryptographic sources, e.g., through proxy-based designs [19], but do not study MKR-style reinforcement in detail.

Possible integration points for MKR keys are hybridization approaches in the sense of combining several key exchange mechanisms. Hybrid key exchange on the level of classical and PQC cryptographic primitives is being standardized in IKEv2, enabling combinations of classical, post-quantum, and pre-shared key mechanisms by (1) dynamically negotiating available PSKs during `IKE_SA_INIT` [20], (2) defining `IKE_INTERMEDIATE` as protocol step to carry the large key exchange data of PQC algorithms [21], (3) using PSKs in `IKE_INTERMEDIATE` [22], (4) enabling multiple key exchanges during `IKE_SA_INIT` and introducing `IKE_FOLLOWUP_KE` for rekeying [23]. All these proposed standards are scrutinized by cryptographic research such as [24]. Similar, hybridization has also been explored in QKD deployments, combining QKD with PQC [25] and multipath solutions [26], [27]. Recent work on opportunistic rekeying provides a formal foundation for combining multiple potentially compromised key materials over time, showing that security can be maintained as long as at least one input remains unknown to the adversary [28].

Taken together, these two research areas reveal a fundamental gap: While multipath techniques have been studied in some system contexts, there is no distributed MKR protocol tailored to structured VPN overlays, nor an evaluation of its security benefits. In particular, it remains open how MKR behaves under realistic routing constraints and hybrid key exchange settings, and how its security gain can be quantified in such environments.

V. Design of a Distributed MKR Protocol

We propose to implement MKR in the defined communication model as described in this section. For this, we first describe how new MKR key material is periodically generated and sent to the correct recipient using randomized routing (combined with backward learning). Subsequently, we present how MKR key materials can be combined over time in a synchronized and authenticated way. Last but not least, we discuss two options how to use the combined key material to reinforce VPN connections.

A. Generation and Routing of Key Material

To keep the overall protocol as simple as possible, we propose the following approach: For each undirected VPN connection $e = \{u, v\} \in E$, we deterministically select an *initiator* and a *responder* for every MKR exchange for e, based on the identifiers u, v. To avoid asymmetries in the number of connections as initiator, we compare the hash values $h(u, v)$ and $h(v, u)$, using u as initiator if the former is smaller. Without loss of generality, let u denote the *initiator* and v the *responder* of an MKR exchange in the remaining section.

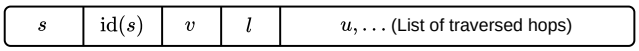

Fig. 2. Structure of an MKR message, which includes the key material s together with its random $\mathrm{id}(s)$, the destination v, the desired number l of *random* intermediate hops, and the path that the message travelled so far. Initially, this only includes the initiator u.

Further, we only use $k = 1$ path for each MKR exchange, which greatly simplifies the synchronization protocol presented in Sec. V-B: Whenever v receives an MKR key material, he can immediately be sure that u also knows the key material and initiate the corresponding synchronization. Similarly to path hopping approaches, the *multipath* idea is therefore realized over time due to periodic execution (see requirement F1). In practice, one could still perform $k' > 1$ MKR exchanges in parallel to mimic the original idea more closely without modifying the synchronization protocol. However, for the scope of this article, we stick to the simple model that MKR is solely performed sequentially, but periodically with a rate of λ for each connection.

Accordingly, an MKR exchange is initiated by u by first generating the actual key material s using an appropriate pseudo-random number generator (requirement S2), and a random key material $\mathrm{id}(s)$. Then, u generates an *MKR message* as depicted in Fig. 2, and sends it to a random neighbor $x \in N(w) \setminus \{v\}$.

At each subsequent hop w, the MKR message m is forwarded according to a two-phase routing protocol:

1) In phase *MKR-1*, m is forwarded to a random neighbor of the current hop w, while avoiding cycles: Let $P(m)$ be the set of hops that m traversed already (its current *path*). Then, the next hop is selected uniformly at random from $N(w) \setminus P(m)$. Two special cases are considered: If the message arrives at its final destination v by chance, it is not forwarded further. If the message is in a dead end, i.e., $N(w) \setminus P(m) = \emptyset$, phase *MKR-2* is activated.
2) In phase *MKR-2*, m is forwarded based on local routing information collected by *authenticated backward learning* during the synchronization protocol (as discussed in Sec. V-B). If w does not yet know how to route to v, the message is dropped (MKR exchange aborted).

If v is not found during *MKR-1*, *MKR-2* is also activated after a random number l of intermediate hops, which is drawn uniformly at random out of $\{1, \ldots, l_{\max}\}$ by the initiator, with a configurable maximum path length during *MKR-1* of $l_{\max}$. In both phases, the current hop w appends itself to the list of traversed hops, so v can see the complete path of the message.

Note that the MKR message is always protected hop-by-hop by the underlying VPN connections. This is crucial to maximize the *effective path diversity*: In the worst-case, path diversity in the actual transport network could be non-existent, e.g., when all packets between VPN nodes are routed via the same Internet exchange point. Then, the cost for an attacker to observe all MKR messages would be low. However, he also

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

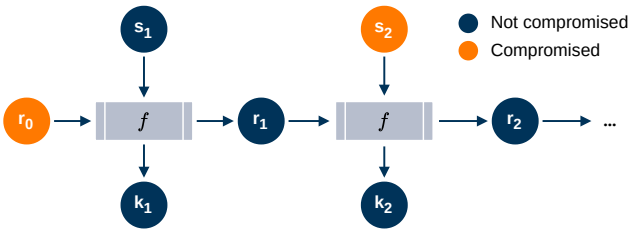

Fig. 3. Starting from an arbitrary initial state r_0, opportunistic re-keying combines a series of input key materials $s_j, 1 \leq j \leq i$ over time by chaining a KDF f [28, Fig. 1]. In our case, the inputs are MKR key materials, and the output keys k_j are used to reinforce the corresponding VPN connection.

has to be able to decrypt messages from various independent VPN connections, increasing the *effective* path diversity.

In summary, using a randomized and distributed routing protocol scales well (requirement N3), and no single node is able to manipulate the routing for a disproportional number of MKR key exchanges (requirement S5).

B. Synchronized and Authenticated Combination

To combine multiple MKR key materials $s_j, 1 \leq j \leq i$ over time, we propose to use the concept of *opportunistic re-keying* as studied in [28] and depicted in Fig. 3 (requirements F1 and S1). There is one such *key chain* for every connection $e = \{u, v\}$. This construction has the desired property called *resilience* in [28] that a single secure input key s_j not known to the attacker is sufficient so that all following output keys $k_i, i \geq j$ cannot be reconstructed. However, we still require an authenticated protocol to keep the key chain in sync between u and v, and to reinforce the VPN connection with the most recent output key k_i (requirements F2 and S3).

Our synchronization protocol is shown in Fig. 4, omitting intermediate hops the messages take. Possible options for routing are discussed subsequently. When the responder v receives an MKR message, he generates a synchronization request for $\mathrm{id}(s)$. This request also includes:

- The discovered path p.
- The next index $i+1$ of the key chain.
- A sequence number n that is incremented for each request for the same index $i+1$. This might be a retransmission for the same key material, or if synchronization fails (no response after retransmissions), also for a different one.
- A digital signature, denoted $\sigma_v(\dots)$, that authenticates v and also proves that it knows the correct value of s and the current state r_i of the key chain. Management of public keys depends on the underlying VPN protocol and is out of scope for this article. For example, in systems based on a public key infrastructure (PKI), the first two messages (MKR packet and synchronization request) could indicate whether public keys and certificates are still cached.

If v receives multiple MKR messages from u, it *must not send* multiple synchronization requests in parallel. Instead, it queues the key materials for later synchronization.

Upon receiving a synchronization request, u temporarily calculates the next state and output key of the key chain,

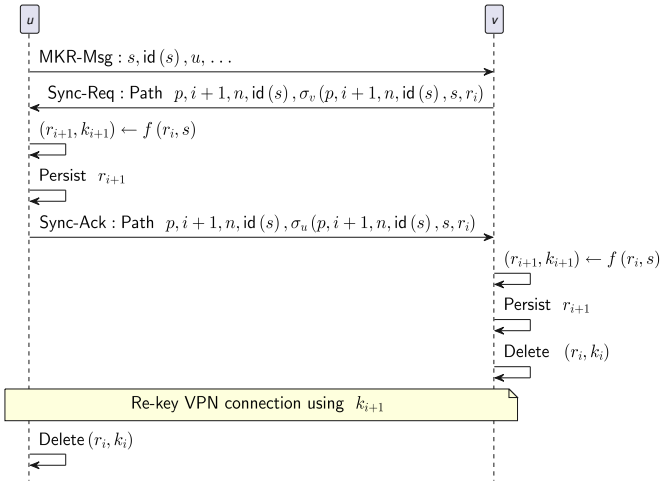

Fig. 4. Authenticated synchronization protocol (inspired by [19, Fig.7]) to incorporate the MKR key material s into opportunistic re-keying, and use the new output key k_{i+1} to reinforce the VPN connection. Signatures by node $x \in \{u, v\}$ over a set of values are denoted by $\sigma_x(\dots)$. Verification of signatures and intermediate hops are not shown for brevity.

but does not delete the old ones yet to allow for a roll-back if required. After persistently storing the (temporary) next state, u sends a synchronization acknowledgment, also including a signature for authentication and to prove that it indeed knows s and r_i. The acknowledgment also includes the discovered path again (for backward learning, see below). Unexpected synchronization requests (wrong index $i+1$, or old sequence number n) are ignored by u. If a new synchronization request for the same index $i+1$ is received, the temporary next state and output key are updated.

Upon receiving the expected synchronization acknowledgment (correct $i+1$ and n), the responder v calculates the next state r_{i+1} and output key k_{i+1} of the key chain and deletes the previous ones. Last but not least, v initiates a re-key of the underlying VPN connection, reinforced by key k_{i+1}. The details of this last step depend on the specific VPN protocol. For example, in the context of IPsec, a recent extension allows using k_{i+1} as a *post-quantum preshared key* (PPK) during IKEv2 re-keying [22], using the index $i+1$ as PPK ID. If the VPN connection is completely re-established in the meantime, k_{i+1} should also be used to reinforce the initial key exchange (also possible with [22]). Another alternative that covers both cases is to use k_{i+1} to completely encrypt all IKEv2 packets using a proxy as in [19]. The initiator u eventually deletes the old state r_i when he can be sure that the responder received his acknowledgment. Instead of using another explicit acknowledgment from v to u, it can be implicit: When receiving the sync request for index $i+2$, or by monitoring whether the VPN re-key was successful.

Regarding the *routing* of the synchronization request and acknowledgment: Both are routed via the same path as the original MKR message. While this increases the bandwidth overhead due to the transmission of potentially large signatures over a longer path than required, it prevents a simple attack on

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

availability (requirement S4): If the messages would be sent on the direct connection $e = \{u, v\}$, attackers could, in the worst case, know both the current session key of e, and the current output key of opportunistic re-keying (i.e., no successful MKR yet). Hence, they could selectively prevent the synchronization of MKR key material that is unknown to them.

Furthermore, using the discovered path again allows using the synchronization request and acknowledgment for *backward learning* within the intermediate hops w: When hop w receives a message of either type from a previous hop w', he learns that he can reach the sender of the message, i.e., u or v, via w', together with the number of hops required. Consequently, he can keep track of the currently shortest path to all endpoints of MKR exchanges. Because both messages also include the complete path information, signed by u or v, respectively, routing information cannot be manipulated by malicious nodes. Attackers can also not forge a non-existent route in the original (not yet authenticated) MKR message: This would be detectable by honest nodes during the processing of the synchronization request and response. To react to teardown of links, a node that has to drop an MKR message during phase *MKR-2* can propagate this information back on the path.

The re-key exchange of the underlying VPN protocol can also be tunneled via the indirect path when using a proxy-based approach [19]. Otherwise, existing implementations simply execute the protocol via the direct link, in which case failed re-keys should be monitored.

VI. Quantitative Evaluation

This section presents a simulation-based evaluation of the proposed MKR protocol in different scenarios.

A. Simulation Setup

We simulated our MKR protocol using the `omnetpp` discrete event simulation framework [29]: Given an overlay graph $G = (V, E)$, each node $v \in V$ is realized by a `cSimpleModule`. Connection properties like data rate and propagation delay are not simulated. Similar, the runtime of cryptographic operations is not simulated. In result, each MKR exchange is "instant". This simplified model is adequate because practical end-to-end latencies and runtimes of cryptographic operations are negligible compared to the evaluated MKR intervals $1/\lambda \geq 2\,\mathrm{min}$ for each connection.

For the threat model, we assume the worst-case scenario for which MKR is still able to secure all connections over time: Given a connected overlay graph $G = (V, E)$ with $n = |V|$ nodes, only $n - 1$ connections are not compromised at simulation start. These $n - 1$ connections form a *spanning tree* of G, i.e., there exists exactly one secure path between any pair of nodes. For our simulations, we randomly determine such a spanning tree. Apart from the secure spanning tree, we assume MKR to be the only secure key exchange. In practice, a secure spanning tree could, e.g., exist due to PSKs, even if the used PQC KEM turns out to be insecure.

Our main research question is: **How long does it take for MKR to converge, i.e., secure every connection?** Apart from this *time to convergence*, we also evaluate the *average bandwidth overhead* introduced by MKR for nodes (requirement N1). Here, we also assume the worst-case, namely hybrid signatures based on SLH-DSA-SHAKE-256f [9], having the largest (standardized) signature size of $49\,856\,\mathrm{B}$, combined with ed448 ($114\,\mathrm{B}$ signatures). Using such large signatures, the remaining overhead of MKR is negligible and not included in measurements. Apart from simulations, we perform a benchmark of SLH-DSA-SHAKE-256f on real hardware to assess the computational overhead (requirement N2).

TABLE I
Simulation parameters

Symbol	Parameter	(Default) values
	Topology	Chord, BA
m	BA graph parameter	6, 12, 18
n	Number of nodes	100, 300, 500, 700, 900
$1/\lambda$	MKR interval	2, 4, 6, 8, 10 min
$l_{\max}$	Maximum random path length	3, 4, 5, 6, 7

As overlay topologies, we compare random, scale-free graphs according to the *Barabasi-Albert (BA) model* (with varying parameter m), to a structured *Chord ring* [30] (as one example for practical VPN overlays [31]). For the BA model, we use the implementation in [32], with a slight modification to avoid the first connection of a newly added node to be a loop (to guarantee G being connected).

Table I summarizes the simulation parameters and their values. For each combination of parameter values, we performed 32 independent simulation runs. Result figures show the mean time to convergence across the 32 runs, and the standard deviation as error bars. Lines connecting the average values are for visual aid and do not indicate continuous measurements.

B. Time to Convergence

The time to convergence in various scenarios is shown in Fig. 5. The main insights are: First, a higher frequency for MKR with shorter time between MKR executions results in a faster time to convergence, independent of the topology, see Fig. 5 (a). Still, it differs across topologies: Especially, the time to convergence increases when there are more connections in total (larger values for m in the BA model). This is because the fraction of initially secure connections is $n-1/|E|$, and thus lower if there are more connections. Then, the initial probability to find a *secure path* is also lower. The Chord rings and BA graphs with parameter $m = 6$ all have a total of $|E| \approx 600$ connections on average. Still, time to convergence is slightly lower for Chord rings, which can be explained by its structure: For each pair of nodes, there are roughly the same number of alternative paths. In contrast, using the BA model with $m = 6$, some nodes can also have a very low degree. This inherently limits the set of available alternative paths with up to $l_{\max}$ hops between some node pairs. For some initially compromised links, this means that they require very specific links to be secured by MKR first before they can be secured themselves. On top, backward

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

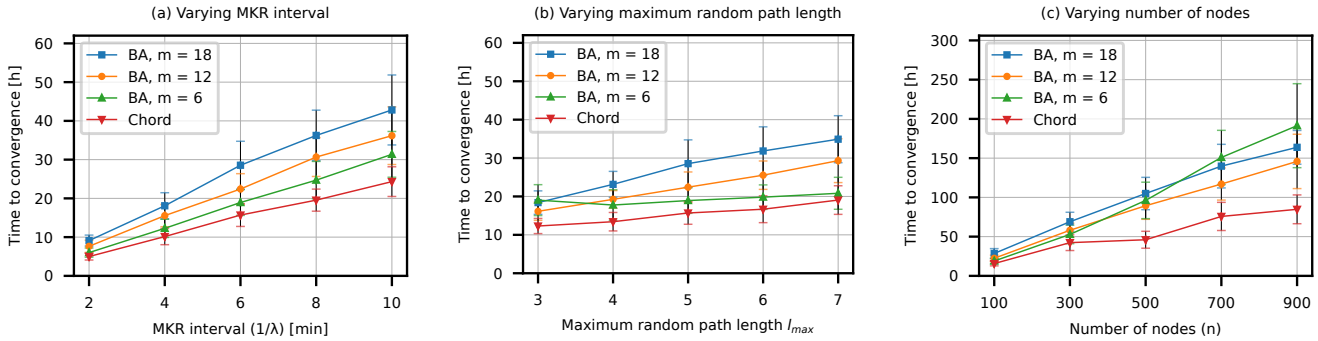

Fig. 5. MKR time to convergence (all VPN connections secure) in the studied scenarios.

learning to nodes with a low degree is not as robust: We observed the most aborted MKR exchanges (no route to the destination found) for BA graphs with $m = 6$ (up to $12\,\%$ of exchanges in this experiment). Still, the time to convergence is lower compared to graphs with more links (BA, $m > 6$).

Second, see Fig. 5 (b), using shorter random path prefixes on average is beneficial for MKR in most scenarios. This can be explained by the fact that the probability that all links on a path are secure decreases with each additional link. But again, BA graphs with $m = 6$ are an exception: Here, increasing the value of $l_{\max}$ counteracts both of the problems described in the previous paragraph. Especially, the ratio of aborted MKR exchanges decreases to $5\,\%$ on average for the BA model with $m = 6$, combined with $l_{\max} = 7$. Consequently, using larger values for $l_{\max}$ is generally the more robust option.

Third, for all studied topologies, the total number of links increases with the number of nodes n. Consequently, the time to convergence of MKR also increases as discussed above and as shown in Fig. 5 (c). For BA graphs with $n = 900$ nodes, combined with parameter $m = 6$, we observed times to convergence of up to ten days. Fortunately, the time it takes to secure a majority of links is lower, see Fig. 6. The observed "S-shaped" curve for the number of secure connections over time can also be explained by the random nature of our MKR protocol: Initially, only few secure paths exists in our worst-case threat model and most MKR exchanges use compromised paths (and can thus not increase security). After this initial phase, a self-reinforcing effect can be observed: The more connections are secured by MKR, the more likely it gets that subsequent MKR exchanges also use secure paths and can secure more connections. In a last phase, the progress to secure the very last connections slows down. Further, Fig. 6 also supports the previous discussion that for BA graphs with $m = 6$, the time to convergence is mainly dominated by a few node pairs (especially slow last phase).

C. Overhead

The average bandwidth overhead of MKR for a node is proportional to its degree, the rate λ on each link, and the size of signatures. Consequently, the largest average overhead per node we observed across all simulation runs was for $n = 100$, using the BA model with $m = 18$, and an MKR interval

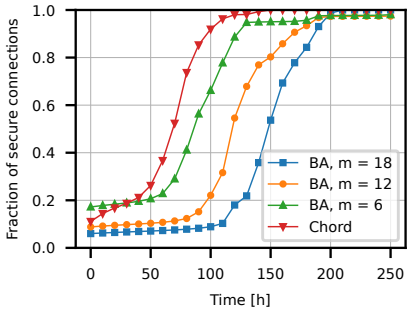

Fig. 6. Fraction of secure VPN connections over time, for $n = 900$ nodes. For each point in time, the figure shows the mean over 32 runs, error bars not shown for visibility. Note that the time of intersection with $y = 1$ does not match the averages in Fig. 5 (c), $n = 900$, because it requires *all* runs to have converged.

of $1/\lambda = 2\,\mathrm{min}$, resulting in $\approx 401\,\mathrm{kbit\,s^{-1}}$. In a vast majority of runs, the overhead was less than $200\,\mathrm{kbit\,s^{-1}}$. In most modern scenarios, this overhead may be assessed as *low*. Still, λ could further be tuned for individual nodes if required, e.g., in case of lower available bandwidth on satellite links. Furthermore, λ could also be tuned dynamically:

1) To speed up convergence in worst-case scenarios, λ could initially be higher and tuned down over time. For example, when initially deploying MKR in a large existing VPN, or when a new node is deployed.
2) The rate could be tuned according to the actual load on a connection or node.

Using the official benchmark script[1] for the SLH-DSA-SHAKE-256f reference implementation (formerly known as SPHINCS+) on an AMD Ryzen 7 5700G CPU, we get the following average results: $114\,\mathrm{ms}$ for signing, $4\,\mathrm{ms}$ for verifying. If required, nodes with many links can also dynamically decrease their computational overhead via λ.

VII. Conclusion

Due to the quantum threat, hybridization in cryptography is more important than ever. In this article, we studied MKR as an additional building block for quantum-resistant VPNs. For

[1] https://github.com/sphincs/sphincsplus/blob/master/benchmark.py

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/

this, we presented a simple, scalable, distributed protocol to transmit MKR key materials and accumulate them over time in a synchronized and authenticated way. Our simulation-based evaluation shows that even under a worst-case threat model, MKR is able to quickly propagate security to a majority of connections in the VPN.

Further, assuming a more realistic threat model, namely that currently deployed PQC KEMs cannot be broken immediately, MKR forces attackers to permanently observe and store the traffic on most links in a VPN, even if they only want to preserve their chances for "store now, decrypt later" attacks on some links or for some periods of time. At the same time, the average overhead for nodes is low and can further be tuned to the conditions at each individual node. In result, MKR is a very cost-effective approach to further fortify hybrid cryptography and quantum-resistance in communication infrastructures.

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997, DOI: 10.1137/S0097539795293172.

[2] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," *Quantum Information and Computation*, vol. 3, no. 4, pp. 317–344, 2003.

[3] National Institute of Standards and Technology, "Module-lattice-based key-encapsulation mechanism standard," 2024," DOI: 10.6028/NIST.FIPS.203.

[4] S. Ehlen, H. Hagemeier, T. Hemmert, S. Kousidis, M. Lochter, S. Reinhardt, and T. Wunderer, "Quantum-safe cryptography – fundamentals, current developments and recommendations," 2022. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Brochure/quantum-safe-cryptography.html?nn=916626

[5] European Union Agency for Cybersecurity, "Post-quantum cryptography: Current state and quantum mitigation," 2021," DOI: 10.2824/92307.

[6] French Cybersecurity Agency, Federal Office for Information Security, Netherlands National Communications Security Agency, and Swedish National Communications Security Authority, Swedish Armed Forces, "Position paper on quantum key distribution," 2024. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Crypto/Quantum_Positionspapier.html

[7] D. Schatz, H. Koerfgen, and G. Schaefer, "Automated distribution of out-of-band key material in virtual private networks," in *Proceedings of the 12th International Conference on Information Systems Security and Privacy*, 2026, pp. 376–387, DOI: 10.5220/0014459200004061.

[8] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)," 2014. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7296

[9] National Institute of Standards and Technology, "Stateless hash-based digital signature standard," 2024," DOI: 10.6028/NIST.FIPS.205.

[10] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983, DOI: 10.1109/TIT.1983.1056650.

[11] D. Dolev, C. Dwork, O. Waarts, and M. Yung, "Perfectly secure message transmission," *Journal of the ACM*, vol. 40, no. 1, pp. 17–47, 1993, DOI: 10.1145/138027.138036.

[12] Y. Desmedt and Q. Yang, *Encyclopedia of Cryptography, Security and Privacy*. Springer Nature Switzerland, 2025, ch. Perfectly Secure Message Transmission, pp. 1790–1793, DOI: 10.1007/978-3-030-71522-9_326.

[13] W. Lou and Y. Kwon, "H-spread: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks," *IEEE Trans. on Vehicular Technology*, vol. 55, no. 4, pp. 1320–1330, 2006, DOI: 10.1109/TVT.2006.877707.

[14] R. Safavi-Naini, A. Poostindouz, and V. Lisy, "Path Hopping: An MTD Strategy for Long-Term Quantum-Safe Communication," *Security and Communication Networks*, vol. 2018, pp. 1–15, May 2018, DOI: 10.1155/2018/8475818.

[15] B. Zhang, H. Li, S. Zhang, J. Sun, N. Wei, W. Xu, and H. Wang, "Multi-Constraint and Multi-Policy Path Hopping Active Defense Method Based on SDN," *Future Internet*, vol. 16, no. 4, p. 143, Apr. 2024, DOI: 10.3390/fi16040143.

[16] D. Lukaszewski and G. G. Xie, "Multipath transport for virtual private networks," in *CSET @ USENIX Security Symposium*, 2017.

[17] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How hard can it be? Designing and implementing a deployable multipath TCP," in *9th USENIX NSDI*, Apr. 2012, pp. 399–412.

[18] X. Xu, S. Hegde, B. Pismenny, D. Zhang, L. Xia, and M. Puttaswamy, "Encapsulating IPsec ESP in UDP for Load-balancing," Internet Engineering Task Force, Internet-Draft draft-xu-ipsecme-esp-in-udp-lb-15, Feb. 2026, Work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-xu-ipsecme-esp-in-udp-lb/15/

[19] D. Schatz, F. Altheide, H. Koerfgen, M. Rossberg, and G. Schaefer, "Virtual private networks in the quantum era: A security in depth approach," in *Proceedings of the 20th International Conference on Security and Cryptography*, 2023, pp. 486–494, DOI: 10.5220/0012121800003555.

[20] S. Fluhrer, P. Kampanakis, D. McGrew, and V. Smyslov, "Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security," Jun. 2020. [Online]. Available: https://www.rfc-editor.org/rfc/rfc8784

[21] V. Smyslov, "Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2)," 2022. [Online]. Available: https://www.rfc-editor.org/rfc/rfc9242.html

[22] ——, "Mixing preshared keys in the IKE_INTERMEDIATE and CREATE_CHILD_SA exchanges of the Internet Key Exchange Protocol Version 2 (IKEv2) for post-quantum security," 2025. [Online]. Available: https://datatracker.ietf.org/doc/rfc9867/

[23] C. Tjhai, M. Tomlinson, G. Bartlett, S. Fluhrer, D. V. Geest, O. Garcia-Morchon, and V. Smyslov, "Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)," RFC 9370, May 2023. [Online]. Available: https://www.rfc-editor.org/info/rfc9370

[24] S.-L. Gazdag, S. Grundner-Culemann, T. Guggemos, T. Heider, and D. Loebenberger, "A formal analysis of IKEv2's post-quantum extension," in *Annual Computer Security Applications Conference*. Virtual Event USA: ACM, Dec. 2021, pp. 91–105, DOI: 10.1145/3485832.3485885.

[25] E. Kalnina, R. Balodis, E. Celms, S. Kozlovics, I. Opmane, K. Petrucena, E. Rencis, and J. Viksna, "Integration of PQC and QKD: Applications, Challenges and Implementation Frameworks," in *Applied Cryptography and Network Security Workshops*, ser. Lecture Notes in Computer Science. Springer Nature Switzerland, 2026, vol. 15653, pp. 266–284, DOI: 10.1007/978-3-032-01799-4_15.

[26] F. Valbusa, T. Lorünser, G. Spini, and S. Laschet, "Relaxing the single point of failure in quantum key distribution networks: An overview of multi-path approaches," in *Availability, Reliability and Security*, ser. Lecture Notes in Computer Science. Springer Nature Switzerland, 2025, vol. 15998, pp. 183–200, DOI: 10.1007/978-3-032-00642-4_11.

[27] J. Li and M. Li, "Multi-Path Secret Sharing for QKD Key Relay in IP Networks," Internet Engineering Task Force, Internet-Draft draft-li-ipsecme-qkd-multipath-secret-sharing-01, Feb. 2026, Work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-li-ipsecme-qkd-multipath-secret-sharing/01/

[28] S. Lucks, D. Schatz, and G. Schaefer, "On the security of Opportunistic Re-Keying," in *Proceedings of the 22nd International Conference on Security and Cryptography*, 2025, pp. 329–338, DOI: 10.5220/0013458000003979.

[29] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the First International ICST Conference on Simulation Tools and Techniques for Communications Networks and Systems*," DOI: 10.4108/ICST.SIMUTOOLS2008.3027.

[30] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001, DOI: 10.1145/964723.383071.

[31] M. Rossberg, G. Schaefer, and T. Strufe, "Distributed automatic configuration of complex IPsec-infrastructures," *Journal of Network and Systems Management*, vol. 18, no. 3, pp. 300–326, 2010, DOI: 10.1007/s10922-010-9168-7.

[32] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnády, "The degree sequence of a scale-free random graph process," *Random Structures & Algorithms*, vol. 18, no. 3, pp. 279–290, 2001, DOI: 10.1002/rsa.1009.

Preprint; accepted and in press at QSNS 2026, https://qubip.eu/qsns2026/