Formalizing PQC Signature Transition: Case of PoW Blockchain Against On-Spend Attack

Kigen Fukuda
kigenf@vt.edu
Virginia Tech
Alexandria, Virginia, USA

Shin'ichiro Matsuo
matsuo@vt.edu
Virginia Tech
Alexandria, Virginia, USA
Georgetown University
Washington, DC, USA

Abstract

It is getting widely recognized that quantum computers pose a fundamental threat to blockchain security. The transaction signature transition to Post-quantum cryptography (PQC) is therefore an urgent challenge. However, it remains unclear how much quantum computing power would be sufficient to compromise blockchain security and, consequently, by when the transition should be completed. To address these questions theoretically, we first formalize the signature transition process and the quantum adversary based on the well-known Bitcoin backbone protocol framework. We then establish a threshold for the chain's tolerable quantum adversary capability. Specifically, we prove that a security property *migration liveness* holds with overwhelming probability if

$$\Delta_{\text{eff}} \geq \left\lceil \frac{4}{(1-\epsilon)f} \right\rceil, \quad \text{and} \quad f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$$

where Δ_{eff} is the number of rounds the quantum adversary needs to produce a forged transaction after the broadcast of a migration transaction, f is the honest mining success probability, ϵ is the concentration quality of the underlying random variables, and κ is the security parameter. We further generalize the analysis to derive a relationship between the transition process and the tolerable quantum adversary capability, providing a theoretical basis for designing secure signature transition plans.

Keywords

Blockchain, PQC, PQC migration, Liveness, PoW

1 Introduction

1.1 Background

The security of current standard digital signature algorithms relies on the computational hardness of solving integer factorization and the elliptic curve discrete logarithm problem (ECDLP). However, it is well known that Shor's algorithm, when run on a quantum computer, can solve these problems in polynomial time. Therefore, a sufficiently powerful quantum computer can compromise such signature schemes by recovering a signing key from a verification key. Recognizing this vulnerability, the need to migrate to post-quantum cryptography (PQC) has been acknowledged across a variety of industries and systems, including blockchain.

Although various blockchain components require cryptographic migration, the most critical concern is the transaction signing algorithm, given its importance and the difficulty of renewing it. Major blockchains, including Bitcoin[27], Ethereum[7], and many others, have relied on ECDSA or EdDSA for transaction signing, both of which are quantum-vulnerable. [1] By deriving the user's signing key from an on-chain-exposed verification key, a quantum attacker can forge valid signatures and execute unauthorized asset transfers, thereby undermining the core principle of blockchain technology: "Not your keys, not your coins." For the rest of the paper, we refer to the transaction signature transition via direct asset transfer to a PQC-derived address as the blockchain PQC migration, excluding other quantum-vulnerable blockchain components and transition methods from the paper's scope.

The impact of quantum attacks on blockchains and the urgency of completing the PQC migration are central questions. Babbush et al. [2] have recently introduced a three-way taxonomy of quantum attacks: *at-rest, on-spend, and on-setup*. [2] In Bitcoin, accounts that have reused addresses or employ P2PK, P2MS, or P2TR formats already expose their verification keys on-chain. In account-based blockchains such as Ethereum, the verification keys of most EOA accounts are inherently exposed. These accounts are vulnerable to *at-rest attacks* in which the attacker can recover the signing key from an exposed verification key without a strict time constraint. On the other hand, accounts whose verification keys are protected by hash functions (e.g., P2PKH with no address reuse) are vulnerable to *on-spend attacks*, in which the attacker can begin computation only after the transaction is broadcast to the network.

Given the tight time constraint in the on-spend setting, the blockchain community views at-rest attacks as the more immediate threat and has focused its migration efforts on preventing on-chain verification key exposure [5, 12]. However, recent work by Babbush et al. [2] and Cain et al. [8] has substantially reduced the estimated quantum resources required to break the 256-bit ECDLP. While the concrete execution time still depends on the physical implementation of the underlying quantum circuit, these improvements suggest that on-spend attacks may become realistic considerably sooner than the community has anticipated.

These developments make rigorous estimation of the timeline for completing blockchain PQC migration increasingly important. Such an estimate requires a formal analysis of the relationship between quantum attack capability and blockchain security. Deriving the necessary quantum computer capabilities to break blockchain security yields the migration deadline by mapping them to the quantum computer development roadmap. To the best of our knowledge,

[1] In the remainder of this paper, we simply refer to any quantum-vulnerable blockchain transaction signing scheme as ECDSA for simplicity.
[2] On-setup attack targets fixed public protocol parameters to produce a reusable backdoor into a protocol, which is out of scope of our analysis.

formal cryptographic analysis of chain-wide PQC migration is limited, and no formal analysis that parameterizes quantum computer signature forgery capabilities against blockchain security has been conducted. Especially, deriving a safe migration timeline requires modeling the worst-case adversarial scenario, which prior work has not addressed.

1.2 Main Contributions

The main contribution of this work is the first rigorous security analysis of the PoW blockchain PQC migration against the quantum adversary. While we recognize the risk of at-rest attacks, we focus exclusively on the on-spend attack setting, as at-rest defenses reduce to address-design that prevent on-chain key exposure. Our analysis formally characterizes the conditions under which a quantum adversary succeeds in an on-spend attack during the PQC migration and derives the quantum computing capability threshold that blockchain security can withstand. Specifically, we provide the following contributions:

(1) **A formal and practical migration model.** We propose a four-phase model for blockchain PQC migration that captures the accounts that cannot or will not migrate before the quantum threat materializes. This model provides a realistic, flexible representation of migration paths.
(2) **Adversarial model with quantum-miner collusion.** To formally analyze the worst-case on-spend attack scenario, we define an adversarial model in which a quantum adversary colludes with adversarial miners to maximize the probability of success. To our knowledge, this is the first analysis that combines quantum forgery capabilities with strategic mining behavior at the consensus layer.
(3) **A sufficient threshold for tolerable quantum adversary capability.** Building on the Bitcoin backbone protocol of Garay et al. [16], we extend their model to the PQC migration setting. We incorporate a quantum adversary's effective capability Δ_{eff} into the execution model, define a security property for the blockchain PQC migration, *migration liveness*, and establish a sufficient condition for its satisfaction. The effective forgery delay Δ_{eff}, which represents the time a quantum attacker requires for forgery, must satisfy
$$\Delta_{\text{eff}} \geq \left\lceil \frac{4}{(1-\epsilon)f} \right\rceil \quad \text{and} \quad f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$$
where f is the honest mining success probability. Once Δ_{eff} falls below this threshold, our sufficient condition no longer applies, and on-spend attack resistance is not guaranteed within our framework.
(4) **Generalized analysis and practical implications.** We generalize the derived threshold for tolerable quantum computer capabilities by parameterizing the robustness of the migration process. We quantify the resulting trade-off between the chain's quantum tolerance, migration robustness, and the maximum tolerable adversarial mining share. These results clarify how chain-specific parameters and migration process design jointly determine the chain's quantum resistance, providing foundational insights for practical migration planning.

1.3 Related Work

As quantum computing research advances, a growing body of research on blockchain PQC migration has emerged. Giechaskiel et al. [19] first provided a rigorous security analysis of the effects of compromised cryptographic primitives on Bitcoin in 2016. Various studies [22, 28] have since extended the scope from Bitcoin to a broader range of blockchain instances, although their analyses remain at an overview level and do not formally model the quantum adversary. A parallel line of work focuses on determining the timeline of the quantum threat to blockchains. Aggarwal et al. [1] analyze the execution time and number of qubits required for breaking Bitcoin security, and Holmes and Chen [20] estimate the necessary execution time to attack various chains. These studies conduct quantitative threat assessments, focusing on physical resource estimation and chain-specific timing constraints, but do not formally model the quantum adversary's capabilities within blockchain execution environments.

Beyond such threat assessment, several works propose concrete blockchain PQC migration mechanisms from both academia and the blockchain community. At the chain-wide level, Sato and Matsuo [29] present the first academic proposal for blockchain signature migration, later refined with a security analysis by Meng and Chen [26]. To the best of our knowledge, [26] remains the only work to formally model the general transition of transaction signature schemes. At the cryptographic-primitive level, proposals divide into at-rest defenses and on-spend defenses. For at-rest defense, the blockchain communities have proposed new address types that eliminate on-chain verification-key exposure or that rotate single-use ECDSA keys [5, 12]. Our work complements this line by formally analyzing the on-spend setting that these proposals deprioritize. For on-spend defense, beyond directly substituting ECDSA with standard PQC signature schemes [23, 24], alternative constructions have been proposed to preserve backward compatibility or enable transition without user intervention, including approaches based on post-quantum zero-knowledge proofs [3, 13], account abstraction [14], and commitment to the legacy secret [21]. While these constructions may enable a seamless PQC migration by avoiding the need for users to transfer assets to the new PQC address system, each carries drawbacks, including residual quantum attack surfaces, computational overhead, or introduces a single point of failure [15].

Most recently, Babbush et al. [2], and Cain et al. [8] provided updated resource estimates for breaking the 256-bit ECDLP. In particular, [2] estimates that a fast-clock cryptographically relevant quantum computer can derive an ECDSA private key in approximately 9 minutes from a primed state under their hardware assumptions, putting the success probability of an on-spend attack against Bitcoin at roughly 41%. While the execution time depends on hardware-specific factors such as gate speed and error-correction cycle time, [2] provides the first analysis showing that on-spend attacks are a near-term rather than distant concern. However, a formal model of the quantum adversary's attack capabilities, particularly its potential collusion with the consensus layer, has yet to be established in the analysis of [2]. A quantum attacker could offer miners a reward to either skip mining the block containing the victim's transaction or exclude the transaction itself, buying additional

time to forge a transaction. Our work provides a formal consensus-layer analysis of scenarios that prior work has not addressed and establishes the threshold for tolerable quantum adversary capability under this sophisticated attack model.

Our security analysis is based on the blockchain execution model rigorously defined by Garay et al. (GKL15), which establishes the security properties for the public transaction ledger [17]. They later refined and extended the analysis in the full version (GKL20) [16], and these frameworks analyze private mining attacks in a discrete-round model. Subsequent research has built on and improved this work by analyzing delays in node communication within the continuous-time model [11, 18]. Also, the work [10] analyzes the quantum effect on the GKL framework, showing that quantum mining yields only a bounded per-query advantage, while the safe settlement time stays the same as in the classical case up to a constant. We primarily use the GKL20 directly as the basis for our analysis because it provides a comprehensive blockchain execution model that includes user transaction broadcasting. Our work extends GKL20 to settings where the underlying signature's unforgeability no longer holds.

1.4 Outline

The remainder of this paper is organized as follows. Section 2 introduces the formal model of blockchain PQC migration and discusses the security of each phase. Section 3 presents the blockchain execution model from GKL20 and our quantum-miner colluding adversary model. Section 4 defines the security property migration liveness, which precludes on-spend attacks and establishes the quantum adversary's condition when migration liveness holds with overwhelming probability. We further develop the analysis by considering the migration robustness. Section 5 discusses the implications of our analysis for the blockchain PQC migration practices. Section 6 concludes.

2 Blockchain PQC Migration Model

2.1 Signature Transition and Dormant Assets

We define the transaction signature transition as the process of transferring assets from an ECDSA-derived address to a PQC-derived address. Therefore, the blockchain PQC migration can be completed safely only if any legitimate migration transaction succeeds without an on-spend attack. Note that the ECDSA-derived address includes only addresses with at-rest protection whose verification key is unexposed, such as unspent UTXOs. Our analysis focuses exclusively on the direct asset transfer approach because this approach most faithfully preserves the blockchain's native trust model without introducing additional cryptographic primitives or single points of failure.

In an ideal scenario, all account owners would complete migration before any quantum computer becomes capable of breaking ECDSA. However, blockchains contain a substantial volume of dormant assets that have not been accessed for extended periods due to reasons such as the loss of signing keys, being locked by long-term contracts (e.g., HTLCs), or prolonged inactivity. [3] These dormant assets are expected to remain unmigrated even in the face of the immediate quantum threat, becoming prime targets for on-spend attacks when their owners eventually attempt migration. [4] Systemic migration plans, such as BIP 361 [25], have been proposed to freeze unmigrated assets, but these approaches have been criticized for compromising the public blockchain's censorship resistance. The uncertainty surrounding dormant-asset migration motivates our model to account for the possibility that ECDSA-signed transactions remain valid even after ECDSA is compromised.

2.2 Four-Phase Migration Model

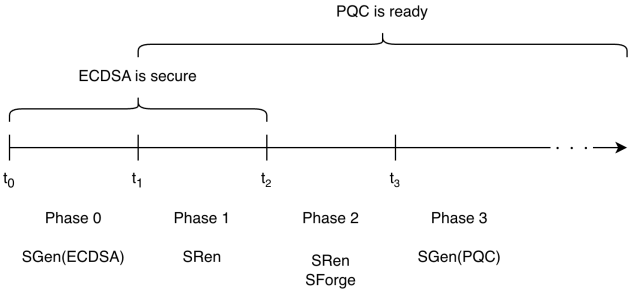


Figure 1: Four-phase migration model. The horizontal axis represents time. ECDSA remains secure throughout Phases 0–1, and secure PQC addresses become available from Phase 1 onward. The algorithms SGen, SRen, and SForge indicate which signing algorithms are active in each phase. Phase 2 is the period in which both legitimate migration (SRen) and quantum forgery (SForge) coexist.

The closest prior formulation, by Meng and Chen [26], describes signature migration in three phases. We extend this to a *four-phase model* (Fig. 1) that explicitly separates the period during which ECDSA-signed transactions remain valid even after ECDSA is broken. This separation is essential for capturing the dormant-asset scenario discussed above, and it allows us to localize the security analysis to a single critical phase. The migration scheme is driven by three algorithms, SGen, SRen, and SForge, corresponding to standard signature generation, signature transition, and quantum signature forgery, respectively.

- **Phase 0 ($t \in [t_0, t_1]$):** Let ECDSA be a EUF-CMA secure signature scheme. Signed transaction tx_{ecdsa} is generated by the algorithm SGen using ECDSA. This is the current state of most blockchain systems.
- **Phase 1 ($t \in [t_1, t_2]$):** ECDSA is threatened but still secure; let PQC be ready, meaning the secure PQC signature scheme and PQC-derived addresses are deployed on-chain and ready to use. Account owners begin issuing migration transactions, $tx_{migrate}$, generated by the SRen algorithm.
- **Phase 2 ($t \in [t_2, t_3]$):** A quantum computer physically capable of breaking ECDSA is realized (e.g., a device with a

[3] For Bitcoin, it is estimated that 20% of the total supply is dormant [6, 9].

[4] Dormant assets include both accounts whose verification keys are already exposed on-chain (e.g., P2PK outputs or address-reused accounts) and those whose keys remain protected. The former are prime targets for at-rest attacks, while the latter become targets for on-spend attacks only when their owners eventually attempt migration. This paper scopes out at-rest vulnerable and unmigratable assets and focuses exclusively on the on-spend setting.

sufficient number of qubits to run Shor's algorithm for 256-bit ECDLP). ECDSA is no longer secure, but the blockchain still accepts the transactions signed with ECDSA. Legitimate migration transactions $tx_{migrate}$ generated by SRen and quantum-forged transactions tx_{forge} generated by SForge *coexist*.

Phase 3 ($t \geq t_3$): Migration is complete either by $tx_{migrate}$ or tx_{forge}, and all users use PQC-derived addresses at this phase. All ECDSA-derived addresses have been either migrated or invalidated by the protocol. Transaction tx_{pqc} is generated by the algorithm SGen using PQC.

This four-phase model successfully captures the unmigrated asset problem and allows flexibility in representing various migration paths by adjusting the duration of Phase 2. If the chain decides to freeze unmigrated assets immediately after ECDSA is compromised, Phase 2 can be eliminated.

2.3 Migration Model Definitions

Throughout this paper, we adopt a simplified transaction model in which each address is controlled by a single signing key. We first introduce three oracles that abstract the cryptographic capabilities available to honest users and the quantum adversary, then define the three algorithms SGen, SRen, and SForge that drive the migration scheme across the four phases.

Transaction generation oracle Txgen**.** Following the formulation of GKL20, we abstract honest user behavior through a stateful oracle Txgen which internally manages key pairs (vk_σ, sk_σ) for user addresses. The oracle exposes the following query:

$$tx \leftarrow \mathsf{Txgen}(1^\kappa, \tilde{tx}),$$

which takes as input an unsigned transaction string $\tilde{tx}$ whose origin address is maintained by the oracle, and returns either a signed transaction tx or $\perp$ if the input is malformed. Each address is permitted only a single transaction issuance. Transaction strings $\tilde{tx}$ are structured as asset transfers between addresses $\{a_\sigma^{origin} \to a_\sigma^{dest}\}$. Signed transactions tx are structured as $\{\varsigma, vk_\sigma, \tilde{tx}\}$, where ς is the valid signature and vk_σ is the verification key of the origin address a_σ^{origin} to verify ς.

Signing key derivation oracle Oshor**.** This oracle abstracts the quantum adversary's key recovery capability. On input of an ECDSA verification key, it returns the corresponding signing key:

$$sk_{ecdsa} \leftarrow \mathsf{Oshor}(1^\kappa, vk_{ecdsa}).$$

Oshor is accessible to the adversary only in Phase 2, reflecting the setting in which the quantum computer becomes capable of breaking ECDSA unforgeability at the start of Phase 2. The detailed timing model of Oshor is deferred to Section 3.8.

Transaction forgery oracle Txforge**.** This stateless oracle models the adversary's ability to sign transactions using externally supplied signing keys sk_σ, thereby bypassing Txgen's key management. The query is:

$$tx \leftarrow \mathsf{Txforge}(1^\kappa, sk_\sigma, \tilde{tx}),$$

which returns a signed transaction structured identically to those issued by Txgen, or $\perp$ if $\tilde{tx}$ is malformed. Therefore, tx generated by Txforge is indistinguishable from that generated by Txgen at the protocol level.

We now define the three algorithms that operate in the four phases of the migration model.

SGen: The signature generation algorithm SGen is implemented in phases 0 and 3. It outputs a signed transaction tx_σ using either ECDSA or a PQC scheme. The generation process of tx_σ is listed as follows:

(1) A user generates $\tilde{tx}_\sigma$ of asset transfer between addresses structured as $\{a_\sigma^{origin} \to a_\sigma^{dest}\}$ for $\sigma \in \{ecdsa, pqc\}$.
(2) The user forms the signed transaction: $tx_\sigma \leftarrow \mathsf{Txgen}(\tilde{tx}_\sigma)$
(3) The user broadcast tx_σ to the blockchain network.

SRen: The signature renewal algorithm, SRen, occurs in Phases 1 and 2. It outputs an ECDSA signed transaction $tx_{migrate}$. This algorithm means that the user issues a transaction to transfer assets from an ECDSA-derived address to a PQC-derived address.

(1) A user generates $\tilde{tx}_{migrate}$ of asset transfer between addresses structured as $\{a_{ecdsa}^{origin} \to a_{pqc}^{dest}\}$.
(2) The user forms the signed transaction: $tx_{migrate} \leftarrow \mathsf{Txgen}(\tilde{tx}_{migrate})$
(3) The user broadcast $tx_{migrate}$ to the blockchain network.

SForge: The signature forgery algorithm, SForge, is executed in Phase 2. It takes a transaction $tx_{migrate}$ as input and outputs a transaction tx_{forge}. This algorithm enables a quantum adversary to forge a transaction from a user's verification key vk_{ecdsa} and transfer coins to the attacker's PQC-derived address.

(1) A quantum adversary $\mathcal{A}_q$ fetch vk_{ecdsa} from $tx_{migrate}$.
(2) $\mathcal{A}_q$ generates a string $\tilde{tx}_{forge}$ of asset transfer between addresses structured as $\{a_{ecdsa}^{origin} \to a_{pqc}^{adv}\}$, where a_{pqc}^{adv} is the $\mathcal{A}_q$'s PQC-derived address.
(3) $\mathcal{A}_q$ computes sk_{ecdsa} from vk_{ecdsa} by querying Shor's oracle: $sk_{ecdsa} \leftarrow \mathsf{Oshor}(vk_{ecdsa})$.
(4) $\mathcal{A}_q$ forms the signed transaction using the forged sk_{ecdsa}: $tx_{forge} \leftarrow \mathsf{Txforge}(sk_{ecdsa}, \tilde{tx}_{forge})$
(5) $\mathcal{A}_q$ broadcast tx_{forge} to the blockchain network.

2.4 Blockchain Security of Each Phase

We now characterize the security of the migration model in each phase. First, we call two signed transactions tx_1 and tx_2 are *conflicting* if $tx_1 \neq tx_2$ and they share the same origin address a_σ^{origin}. Conflicting transactions cannot coexist in any valid chain; the formal description of this constraint is deferred to Section 3.7. Oracle Txgen is said to be *unambiguous* if, for every PPT adversary and every transaction tx issued by Txgen, the probability that the adversary produces a conflicting transaction is negligible in the security parameter κ. This unambiguity property is critical to ensuring that the transaction issued by Txgen is recorded on the chain without interference from a conflicting transaction.

Phases 0, 1, and 3: unambiguity from signature unforgeability. Recall the security assumptions of each phase. In Phases 0, 1, and 3, the underlying signature scheme is assumed to be EUF-CMA secure. The unambiguity of Txgen then follows directly: Txgen doesn't produce two transactions with the same origin address. Any conflicting transaction tx' produced without Txgen by an adversary must carry a valid signature of the shared origin address a_σ^{origin}; since the corresponding signing key is held inside Txgen and never

exposed to the adversary, producing such a signature would constitute a successful forgery against the underlying scheme, which occurs with negligible probability.

Phase 2: unambiguity broken by Oshor **and** Txforge**.** The quantum adversary $\mathcal{A}_q$ gains access to Oshor to recover sk_{ecdsa} from Txgen generated transaction $tx_{migrate}$ in phase 2. Combined with Txforge, the adversary can produce a signed transaction tx_{forge} that conflicts with a previously issued $tx_{migrate}$ for the same origin address a_{ecdsa}^{origin}. The unambiguity of Txgen therefore no longer holds in Phase 2.

The loss of unambiguity of Txgen gives the adversary the capability to produce a conflicting forged transaction, but this alone does not constitute a successful on-spend attack. Two conflicting transactions cannot coexist in any valid chain, so $tx_{migrate}$ and tx_{forge} compete for inclusion: whichever is recorded on-chain determines the destination of the assets in address a_{ecdsa}^{origin}. The on-spend attack succeeds only if tx_{forge} outpaces $tx_{migrate}$ and is recorded in the chain. Whether such outpacing is feasible depends on the time required for Oshor to recover the signing key and the consensus-layer dynamics, including the behavior of malicious miners. Phases 0, 1, and 3 inherit the security of the underlying signature scheme and require no further analysis of tolerance to on-spend attacks. Phase 2 is the only phase in which Txgen loses unambiguity, and is therefore the sole focus of our security analysis. If the PQC migration is shown to resist the on-spend attack throughout Phase 2, the entire migration is secure against the on-spend attack. In the following section, we formalize the blockchain execution model and adversarial model needed to analyze whether tx_{forge} can outpace $tx_{migrate}$.

3 Backbone Protocol and Adversarial Model

In this section, we introduce the model and essential results of the Bitcoin backbone protocol framework of GKL20 [16], which provides the formal foundation for our analysis. The framework abstracts blockchain consensus into a synchronous, round-based execution model, enabling rigorous probabilistic analysis of core security properties of the blockchain protocol, such as liveness and persistence. We present its main definitions and results in a form tailored for our purposes, then extend the model with the quantum adversarial model for Phase 2 of the PQC migration. Table 1 summarizes all parameters used in the analysis.

3.1 Execution Model

The backbone protocol is a consensus-layer protocol analyzed in a synchronous network model. The execution involves n parties $P_1, \ldots, P_n$, of which t may be *corrupted* by adversarial miner $\mathcal{A}_m$ and deviate arbitrarily from the protocol; the remaining $n-t$ parties are *honest* and faithfully follow the protocol. The parties are driven by an *environment* $\mathcal{Z}$ that provides inputs and receives outputs.

The protocol proceeds in discrete *rounds*. In each round, every party receives messages via a *Diffuse functionality*, performs local mining computation, and sends a message via Diffuse. The Diffuse functionality guarantees that honest parties' messages are delivered by the end of the round. This abstraction models peer-to-peer communication as in Bitcoin.

The hash function $H(\cdot) : \{0,1\}^* \to \{0,1\}^\kappa$, together with another hash function $G(\cdot)$ with the same output length, is modeled as an independent random oracle. Every party is allowed exactly q queries to $H(\cdot)$ per round, capturing the party's hashing power. Specifically, a *block* is a triple $B = \langle h_r, x, ctr \rangle$ where $h_r \in \{0,1\}^\kappa$ is the hash reference to the previous block, $x \in \{0,1\}^*$ is the block's content, and $ctr \in \mathbb{N}$ is the proof-of-work nonce. A block is *valid* if it satisfies

$$H(ctr, G(h_r, x)) < T \quad \text{and} \quad ctr \le q,$$

where T is the target difficulty parameter. A party succeeds in mining a block in a given round if it finds parameters satisfying this inequality within its q queries. The adversarial miner $\mathcal{A}_m$, controlling t parties, therefore performs $t \cdot q$ queries per round. We refer to these constraints as the *q-bounded synchronous setting* of parties and $\mathcal{A}_m$.

Table 1: Parameters in our analysis.

Symbol	Meaning
κ	Security parameter; length of hash function output
λ	Tail-bounds parameter (cf. Definition 3.1, 4.6)
n	Total number of mining parties
t	Number of parties controlled by the adversary
T	Target difficulty level for the PoW inequality
q	Maximum number of hash queries per party per round
p	Probability of success of a single hash query ($p = T/2^\kappa$)
f	Probability that at least one honest party finds a PoW in a round
L	The total run-time of the system
δ	Honest majority advantage: $t/(n-t) \le 1-\delta$
ϵ	Concentration quality of random variables in typical executions
k	Depth parameter for common prefix
ℓ	Length parameter for chain quality
μ	Chain quality parameter (minimum ratio of honest blocks)
s	Number of consecutive rounds
τ	Chain growth parameter
Δ_q	Number of rounds for the Shor oracle to recover a secret key
Δ_{eff}	Effective forgery delay; equals Δ_q in the on-spend setting
Δ^*	Tolerable $\mathcal{A}_q$ duration threshold: $\lceil 4/((1-\epsilon)f) \rceil$
ℓ_h	Required number of honest blocks in the migration window
$\Delta^*(\ell_h)$	Generalized tolerable $\mathcal{A}_q$ duration threshold

3.2 Backbone Protocol

A *blockchain* or simply *chain* C is a sequence of blocks in which each block's hash reference h_r equals the hash of the preceding block. The empty string ε is also a valid chain. For a non-negative integer k, we write $C^{\lceil k}$ to denote the chain obtained by pruning the k rightmost blocks. If C_1 is a prefix of C_2, we write $C_1 \preceq C_2$. The *content vector* of a chain C of length m is $\mathbf{x}_C = \langle x_1, \ldots, x_m \rangle$, collecting the x-value from each block in order.

Chain validation. A chain C is called *valid* if (i) every block satisfies the PoW inequality and correct hash-chain linking to its predecessor, and (ii) the content validation predicate $V(\mathbf{x}_C) = 1$ holds, where $V(\cdot)$ is an application-specific predicate that enforces semantic consistency of the block contents (e.g., absence of conflicting transactions). By convention, $V(\varepsilon) = 1$.

High-level protocol description. Each party maintains a local chain C and in every round:

(1) **Receive and validate.** The party checks its RECEIVE tape for incoming chains. Each candidate chain is validated both structurally (well-formed PoW and hash-chain linking for every block) and semantically ($V(\mathbf{x}_C) = 1$). If a valid incoming chain is strictly longer than the party's local chain, the party abandons its local chain entirely and adopts the incoming chain; Otherwise, it retains its local chain.
(2) **Determine input.** The party computes

$$x \leftarrow I(st, C, round, \text{INPUT}(), \text{RECEIVE}())$$

where $I(\cdot)$ is the application-specific input contribution function. The function $I(\cdot)$ is required to produce values consistent with $V(\cdot)$: for any chain C with $\mathbf{x}_C = \langle x_1, \ldots, x_m \rangle$, the output x satisfies $V(\langle x_1, \ldots, x_m, x \rangle) = 1$.
(3) **Proof of work.** The party queries $H(\cdot)$ up to q times, seeking a nonce ctr such that $H(ctr, G(h_r, x)) < T$.
(4) **Diffuse.** If a PoW is found, the party diffuses the extended chain to all parties.

The protocol is also parameterized by a chain reading function $R(\cdot)$ that specifies how a chain is interpreted by the application layer. Since an honest party only holds a chain C that has passed validation in step (1), and only extends it with inputs satisfying $V(\cdot)$ in step (2), the content vector $\mathbf{x}_C$ of any honest party's chain satisfies $V(\mathbf{x}_C) = 1$ at all times.

3.3 Random Variables and Parameters

The central parameter of the protocol is f, the probability that at least one honest party computes a successful PoW in a given round:

$$f \;=\; 1 - (1-p)^{q(n-t)},$$

where p is the probability of success of a single hash query. For each round i, define the following Boolean random variables:

- $X_i = 1$ if at round i at least one honest party obtains a PoW; otherwise $X_i = 0$.
- $Y_i = 1$ if at round i *exactly one* honest party obtains a PoW (a *uniquely successful round*); otherwise $Y_i = 0$.
- $Z_i = \sum_{k=1}^{t} \sum_{j=1}^{q} Z_{ijk}$, where $Z_{ijk} = 1$ if the j-th query of the k-th corrupted party at round i is successful.

For a set of rounds S, let $X(S) = \sum_{r \in S} X_r$, and define $Y(S)$, $Z(S)$ analogously.

3.4 Typical Executions

The probabilistic analysis of the backbone protocol is organized around the notion of a *typical execution*: an execution in which the random variables $X(S), Y(S), Z(S)$ defined above stay close to their expectations over every sufficiently long window of consecutive rounds. The backbone properties stated in Section 3.6 are deterministic consequences of typicality, so all probabilistic guarantees in our analysis ultimately reduce to the probability that an execution is typical.

Definition 3.1 (Typical execution). An execution is typical for $\epsilon \in (0,1)$ and integer $\lambda \geq 2/f$, if for any set S of at least λ consecutive rounds, the following hold:

(1) $(1-\epsilon)\,\mathbb{E}[X(S)] < X(S) < (1+\epsilon)\,\mathbb{E}[X(S)]$ and $(1-\epsilon)\,\mathbb{E}[Y(S)] < Y(S)$.
(2) $Z(S) < \mathbb{E}[Z(S)] + \epsilon\,\mathbb{E}[X(S)]$.
(3) No insertions, no copies, and no predictions occurred. Refer to the original paper [16] for the formal definitions of these events.

The parameter ϵ controls the concentration quality, and λ is the tail-bounds parameter that determines the minimum window length over which concentration holds. Setting $\lambda = \Omega(\log^2 \kappa)$ makes the failure probability negligible in the security parameter. An execution is typical with probability $1 - e^{-\Omega(\epsilon^2 \lambda f + \kappa - \log L)}$.

3.5 The Honest Majority Assumption

The security analysis of the backbone protocol requires a condition on the relative power of honest and adversarial parties. We adopt the following formulation from GKL20.

Definition 3.2 (Honest Majority Assumption). A number t out of n parties are corrupted such that

$$t \;\leq\; (1-\delta)(n-t),$$

where δ satisfies $3f + 3\epsilon < \delta \leq 1$.

This assumption ensures that honest parties produce blocks at a sufficiently higher rate than the adversary, with a margin large enough to absorb the concentration error ϵ from typical executions. The requirement $3f + 3\epsilon < \delta$ implies the need for appropriate PoW difficulty calibrating f under the honest majority advantage δ.

3.6 Backbone Protocol Properties

There are three fundamental properties of the backbone protocol: *chain growth*, *common prefix*, and *chain quality*. These properties are defined as predicates over protocol executions and are shown to hold in any typical execution.

Definition 3.3 (Chain Growth). The *chain growth property* with parameters $\tau \in \mathbb{R}$ and $s \in \mathbb{N}$ states that for any honest party P with chain C, after any s consecutive rounds it adopts a chain that is at least $\tau \cdot s$ blocks longer than C.

THEOREM 3.4 (CHAIN GROWTH). *In a typical execution, the chain growth property holds with parameters $\tau = (1-\epsilon)f$ and $s \geq \lambda$.*

Definition 3.5 (Common Prefix). The *common prefix property* with parameter $k \in \mathbb{N}$ states that for any pair of honest parties P_1, P_2 adopting chains C_1, C_2 at rounds $r_1 \leq r_2$, it holds that $C_1^{\lceil k} \preceq C_2$.

THEOREM 3.6 (COMMON PREFIX). *In a typical execution, the common prefix property holds with parameter $k \geq 2\lambda f$.*

Definition 3.7 (Chain Quality). The *chain quality property* with parameters $\mu \in \mathbb{R}$ and $\ell \in \mathbb{N}$ states that for any honest party P with chain C, for any ℓ consecutive blocks of C, the ratio of honest blocks is at least μ.

THEOREM 3.8 (CHAIN QUALITY). *In a typical execution, the chain quality property holds with parameters $\ell \geq 2\lambda f$ and*

$$\mu \;=\; 1 - \left(1 + \tfrac{\delta}{2}\right) \cdot \frac{t}{n-t} - \frac{\epsilon}{1-\epsilon} \;>\; 1 - \left(1 + \tfrac{\delta}{2}\right) \cdot \frac{t}{n-t} - \frac{\delta}{2}.$$

A direct consequence is that any sufficiently long segment of an honest party's chain must contain at least one honest block.

COROLLARY 3.9. *Any $\lceil 2\lambda f \rceil$ consecutive blocks in the chain of an honest party during a typical execution contain at least one honest block.*

3.7 Public Transaction Ledger

GKL20 instantiates the backbone protocol as a *public transaction ledger*, thereby formalizing the application layer we referenced in Section 2.3. A public transaction ledger is defined with respect to a set of valid ledgers $\mathcal{L}$, a set of valid transactions $\mathcal{T}$, and a transaction generation oracle Txgen that issues transactions on behalf of honest users. Two transactions tx_1, tx_2 are *conflicting* if $C(tx_1, tx_2) = 1$, where $C(\cdot,\cdot)$ is a symmetric relation on $\mathcal{T}$. As defined in Section 2.3, the oracle Txgen is called *unambiguous* if no PPT adversary can produce a transaction conflicting with one issued by Txgen, except with negligible probability in κ.

The three functions $V(\cdot)$, $I(\cdot)$, and $R(\cdot)$ that turn the backbone protocol into Π_{PL}, a protocol that realizes a public transaction ledger, are determined as follows:

- Content validation predicate $V(\cdot)$: $V(\langle x_1, \ldots, x_m\rangle)$ is true if and only if the vector $\langle x_1, \ldots, x_m\rangle$ is a valid ledger, i.e., $\langle x_1, \ldots, x_m\rangle \in \mathcal{L}$.
- Chain reading function $R(\cdot)$: If $V(\langle x_1, \ldots, x_m\rangle) = \mathsf{True}$, the value $R(C)$ is equal to $\langle x_1, \ldots, x_m\rangle$; undefined otherwise.
- Input contribution function $I(\cdot)$: If the input tape contains (Insert, v), it parses v as a sequence of transactions and retains the largest subsequence $x' \preceq v$ that is valid with respect to $\mathbf{x}_C$ (and whose transactions are not already included in $\mathbf{x}_C$).

We record the following corollary for later analysis.

Corollary 3.10 (Chain Consistency). *In Π_{PL}, for any honest party P holding a chain C at any round r, the content vector $\mathbf{x}_C$ satisfies $\mathbf{x}_C \in \mathcal{L}$. In particular, C contains no two transactions tx_1, tx_2 with $C(tx_1, tx_2) = 1$.*

GKL20 introduces two essential properties of such a ledger.

Definition 3.11 (Persistence and Liveness). A protocol Π implements a *robust public transaction ledger* in the q-bounded synchronous setting if it organizes the ledger as a hash chain of blocks of transactions and satisfies:

Persistence (parameterized by $k \in \mathbb{N}$): If at a certain round an honest party reports a ledger containing a transaction tx in a block more than k blocks from the end of the ledger (such a transaction is called *stable*), then tx will be reported by any honest party in the same position in the ledger from this round on.

Liveness (parameterized by $u, k \in \mathbb{N}$): Provided that a transaction tx (either issued by Txgen, or neutral) is given as input to all honest parties continuously for u consecutive rounds, then all honest parties will report tx more than k blocks from the end of the ledger (i.e., all report it as stable).

Lemma 3.12 (Persistence). *Under the Honest Majority Assumption, the public transaction ledger protocol Π_{PL} with parameter $k = \lceil 2\lambda f \rceil$ satisfies Persistence with probability at least $1 - e^{-\Omega(\epsilon^2 \lambda f)}$.*

Lemma 3.13 (Liveness). *Under the Honest Majority Assumption, Π_{PL} with parameters $u = \lceil 4\lambda/(1-\epsilon) \rceil$ and $k = \lceil 2\lambda f \rceil$ satisfies Liveness with probability at least $1 - e^{-\Omega(\epsilon^2 \lambda f)}$.*

The proof of Liveness proceeds as follows: after u rounds, chain growth ensures that the honest parties' chains have grown by at least $2k$ blocks; chain quality then ensures that at least one block in the k-suffix of $C^{\lceil k}$ was computed by an honest party. Since the underlying signature scheme is unforgeable and Txgen is unambiguous, no conflicting transaction exists, so this honest block must contain the target transaction without dropping it. This unambiguous assumption fails in Phase 2, motivating our analysis. We are now ready to describe our analysis built on top of the GKL20 framework.

3.8 The Shor Oracle Timing

In section 2.3, we model the quantum adversary $\mathcal{A}_q$ key-recovery capability as an access to the oracle Oshor. Here, we parameterize the capabilities of a quantum computer by the number of rounds required to obtain the result of Shor's algorithm, denoted by Δ_{q}.

Definition 3.14 (Shor Oracle). The Shor oracle Oshor is a deterministic oracle parameterized by $\Delta_{\mathrm{q}} \in \mathbb{N}$. Given an ECDSA verification key vk_{ecdsa} as input at some round r_{start}, the oracle returns the corresponding secret key sk_{ecdsa} at round $r_{\mathrm{start}} + \Delta_{\mathrm{q}}$.

Shor's algorithm for the ECDLP is a deterministic polynomial-time quantum algorithm whose runtime, for fixed hardware parameters, is determined by the circuit depth, gate speed, and error-correction cycle time [2]. On a fault-tolerant quantum computer satisfying standard error-rate assumptions, the algorithm succeeds with overwhelming probability; we absorb the negligible failure probability into the negligible-in-κ. Modeling Oshor as a deterministic oracle returning sk_{ecdsa} exactly Δ_{q} rounds after its query is therefore a faithful abstraction. Moreover, whatever strategy $\mathcal{A}_q$ adopts, including multi-target parallelized key recovery, the parameter Δ_{q} subsumes it by definition. Δ_{q} denotes the number of rounds until $\mathcal{A}_q$ produces its fastest successful forgery in the given setting, and it is therefore not tied to a single designated key. Capturing the fastest forgery is what our worst-case analysis requires to derive a sufficient threshold.

3.9 Adversarial Model

We now define the adversarial model for Phase 2 of the migration. As we've introduced, the adversary consists of two entities: classical adversarial miners $\mathcal{A}_m$ controlling a subset of mining parties, and a quantum adversary $\mathcal{A}_q$ holding key-recovery capability via Oshor. We treat them separately because they operate at different layers of the protocol: $\mathcal{A}_m$ at the consensus layer through the standard q-bounded synchronous setting, while $\mathcal{A}_q$ at the execution layer through Phase 2. We then specify their collusion capabilities and an additional assumption on parties' transaction selection, which jointly determine the worst-case on-spend attack scenario.

Adversarial miners $\mathcal{A}_m$. The malicious miners $\mathcal{A}_m$ control t out of n parties and follow the adversarial model of the GKL20 backbone protocol. Specifically, $\mathcal{A}_m$ is a rushing adversary that performs $t \cdot q$ hash queries per round, observes all honest parties' messages within a round before deciding its own strategy, and may inject arbitrary messages, reorder delivery, and spoof source addresses through the Diffuse functionality, subject only to the constraint that honest messages cannot be suppressed or altered. We emphasize that $\mathcal{A}_m$ is not restricted to any particular mining strategy and may deviate arbitrarily from the protocol. This generality is inherited

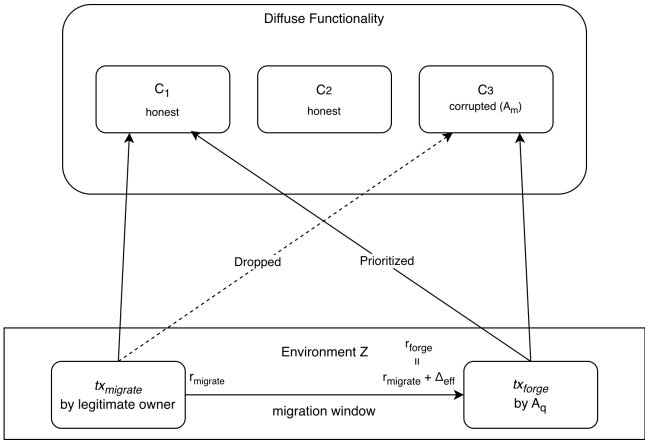


Figure 2: Execution model in Phase 2. Parties $C_1, \ldots, C_n$ operate within the Diffuse functionality; C_3 represents a corrupted miner $\mathcal{A}_m$. At round r_{migrate}, the legitimate owner broadcasts $tx_{migrate}$, which is delivered to all honest parties (solid arrows) but dropped by $\mathcal{A}_m$ (dashed arrow). After Δ_{eff} rounds, the quantum adversary $\mathcal{A}_q$ produces tx_{forge}, which is prioritized for inclusion. The interval $[r_{\text{migrate}},\ r_{\text{forge}} - 1]$ is the migration window during which honest blocks can include $tx_{migrate}$ before the conflicting tx_{forge} becomes available.

from the GKL20 framework and ensures that our model captures any attack strategy, including those targeting specific transactions for exclusion.

Quantum adversary $\mathcal{A}_q$. $\mathcal{A}_q$ executes the SForge algorithm defined in Section 2.3 by interacting with the environment $\mathcal{Z}$ as follows. Let r_{migrate} be the round at which the legitimate user broadcasts $tx_{migrate}$, writing it to $\mathcal{Z}$. From this round, $tx_{migrate}$ becomes accessible to all parties and $\mathcal{A}_q$, and $\mathcal{A}_q$ reads vk_{ecdsa} of the origin address a_{ecdsa}^{origin} from $tx_{migrate}$. $\mathcal{A}_q$ then queries $\text{Oshor}(vk_{ecdsa})$ within the round r_{migrate} and obtains sk_{ecdsa} at round $r_{\text{migrate}} + \Delta_{\text{q}}$. Using the recovered sk_{ecdsa}, $\mathcal{A}_q$ computes the forged transaction $tx_{forge} \leftarrow \text{Txforge}(sk_{ecdsa}, \tilde{tx}_{forge})$ and writes it to $\mathcal{Z}$ at the round $r_{\text{forge}} = r_{\text{migrate}} + \Delta_{\text{q}}$. From r_{forge} onward, tx_{forge} is available in the $\textsc{Input}()$ tape of every party. The signing and broadcasting are completed within the same round r_{forge}.

The effective forgery delay Δ_{eff}, the number of rounds the quantum adversary needs to produce a forged transaction, is therefore $\Delta_{\text{eff}} = \Delta_{\text{q}}$ in our on-spend setting. Although at-rest attacks lie outside the scope of this paper, we remark that they may be covered by a simple extension of the model: if the verification key vk_{ecdsa} is exposed at some round $r_{\text{exp}} \leq r_{\text{migrate}}$, $\mathcal{A}_q$ gains a pre-computation head start of $\Delta_{\text{head}} = r_{\text{migrate}} - r_{\text{exp}} \geq 0$ rounds, reducing the effective delay to $\Delta_{\text{eff}} = \Delta_{\text{q}} - \Delta_{\text{head}}$. We take $\Delta_{\text{eff}} = \Delta_{\text{q}}$ throughout the remainder of the paper, so that $r_{\text{forge}} = r_{\text{migrate}} + \Delta_{\text{eff}}$.

Collusion between $\mathcal{A}_m$ and $\mathcal{A}_q$. In our worst-case adversarial model, we assume the maximal collusion between $\mathcal{A}_m$ and $\mathcal{A}_q$: the quantum adversary $\mathcal{A}_q$ has full control over the block-construction strategy of $\mathcal{A}_m$. Concretely, $\mathcal{A}_q$ may offer a financial incentive to all corrupted parties under $\mathcal{A}_m$, for example, by promising a share of the stolen assets as an out-of-band mining reward, in exchange for following $\mathcal{A}_q$'s directives on transaction inclusion. Under this assumption, $\mathcal{A}_m$ behaves as a strategic instrument of $\mathcal{A}_q$, and in particular:

- $\mathcal{A}_m$ excludes $tx_{migrate}$ from any block it mines, even when $tx_{migrate}$ appears in $\mathcal{A}_m$'s $\textsc{Input}()$ tape.
- Once tx_{forge} becomes available at round r_{forge}, $\mathcal{A}_m$ prioritizes the inclusion of tx_{forge} in every block it mines.

This censorship and prioritization capability extends naturally to alternative adversarial mining strategies: $\mathcal{A}_m$ may, for instance, conduct private mining attacks to grow a private chain which excludes $tx_{migrate}$ and includes tx_{forge} once it becomes available. The full generality of such strategies is already captured by $\mathcal{A}_m$'s arbitrary protocol-deviation capability described above.

Transaction prioritization assumption. Beyond the collusion between adversarial entities, our model also assumes how all parties, including honest ones, select between conflicting transactions. We assume that when a party builds a block and has neither of the conflicting transactions $tx_{migrate}$ and tx_{forge} already committed in its local chain, it adopts tx_{forge} over $tx_{migrate}$. For $\mathcal{A}_m$, this is immediate from the censorship arrangement above. For honest parties, the assumption is realistic because $\mathcal{A}_q$ may bid a substantially higher transaction fee to tx_{forge}, giving every rational miner a direct economic incentive to prioritize tx_{forge}, known as a replace-by-fee bidding. We acknowledge that this assumption deviates from the first-seen rule adopted as a relay policy in legacy Bitcoin. However, the first-seen rule is a soft policy outside the consensus rules and is therefore unenforceable. Actually, miners have a conflicting financial incentive to prioritize high-fee transactions over first-seen transactions. Our model, therefore, takes the position that fee-based prioritization wins.

In sum, our adversarial model captures the strongest realistic on-spend attack capability in the Bitcoin backbone framework. A quantum adversary $\mathcal{A}_q$ capable of breaking ECDSA via Oshor, colluding with malicious miners $\mathcal{A}_m$ that fully control transaction inclusion in their blocks, while honest miners are induced to prioritize the forged transaction at block construction time once it becomes available.

4 Migration Liveness Analysis

4.1 From Liveness to Migration Liveness

Ultimately, we want to derive the condition on $\mathcal{A}_q$ under which the blockchain in Phase 2 maintains on-spend attack resistance with the same probabilistic guarantee as other blockchain security properties: it always holds in typical execution. First, we formalize the definition of an on-spend attack in Phase 2 and show that no on-spend attack can succeed as long as liveness holds for the public ledger protocol Π_{PL}.

Definition 4.1 (On-Spend Attack). Suppose $tx_{migrate}$ is given as input to all parties continuously for u consecutive rounds starting from round r_{migrate}. An on-spend attack succeeds at round $r \geq r_{\text{migrate}} + u$ if tx_{forge} generated by $\mathcal{A}_q$ is reported by any honest party more than k blocks from the end of its chain at round r.

Lemma 4.2 (Liveness Precludes On-Spend Attack). *Suppose that, in a typical execution, liveness holds for Π_{PL} with parameters $u = \lceil 4\lambda/(1-\epsilon)\rceil$ and $k = \lceil 2\lambda f\rceil$. Then no on-spend attack succeeds.*

Proof. Under the assumption that liveness holds, there exists a block that reports $tx_{migrate}$ more than k blocks from the end of its chain. By persistence, which holds in any typical execution and does not depend on the unambiguity of Txgen, $tx_{migrate}$ remains stably reported in the same position by every honest party at every round r. By chain consistency (Corollary 3.10), no honest party's chain can contain both $tx_{migrate}$ and tx_{forge}, since $C(tx_{migrate}, tx_{forge}) = 1$ by construction. Therefore, no honest party reports tx_{forge} more than k blocks deep at any round $r \geq r_{\mathrm{migrate}} + u$. □

Recall that the unambiguity of Txgen fails in Phase 2. As seen in Section 3.7, the proof of liveness relies on the unambiguity of Txgen; the original liveness is not guaranteed in Phase 2. Therefore, to analyze the PQC migration on-spend attack resistance, we introduce a Phase 2 instantiation of liveness and characterize the conditions under which it is satisfied.

4.2 Migration Liveness

We define security property *migration liveness*, an instantiation of the liveness property of Π_{PL} targeting $tx_{migrate}$ in Phase 2, and identify the conditions equivalent to migration liveness satisfaction.

Definition 4.3 (Migration Liveness). The migration liveness property, parameterized by $u, k \in \mathbb{N}$, states the following: provided that $tx_{migrate}$ is given as input to all honest parties continuously for u consecutive rounds starting from round r_{migrate}, then all honest parties will report $tx_{migrate}$ more than k blocks from the end of the chain.

Remark 4.4. Lemma 4.2 remains valid with migration liveness in place of liveness, since it requires only that $tx_{migrate}$ is reported at depth more than k.

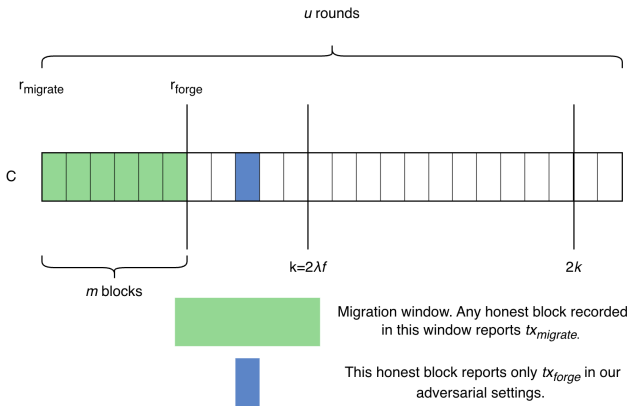


Figure 3: After u rounds, the chain C of any honest party grows by at least $2k$ blocks. Migration liveness holds if and only if the m blocks generated during the migration window $[r_{\mathrm{migrate}}, r_{\mathrm{forge}}-1]$ contain at least one honest block. Any honest block generated after r_{forge} may only contain tx_{forge}.

Lemma 4.5 (Migration Liveness Reduction). *Under the adversarial model of Section 3.9, the following equivalence holds in a typical execution of the public transaction ledger protocol Π_{PL} with parameters $u = \lceil 4\lambda/(1-\epsilon)\rceil$ and $k = \lceil 2\lambda f\rceil$:*

Migration liveness holds if and only if, in the chain C of any honest party after u rounds, there exists at least one honest block generated during rounds r_{migrate} to $r_{\mathrm{forge}} - 1$.

Proof. Let m denote the number of blocks in C that were generated during rounds r_{migrate} to $r_{\mathrm{forge}} - 1$, the window we call the migration window. We consider the state of the chain C of any honest party after u rounds and analyze the two directions separately.

First, we prove that if an honest block exists in C during the migration window, then migration liveness holds. Suppose there exists an honest block in C generated during some round j within the migration window $r_{\mathrm{migrate}} \leq j \leq r_{\mathrm{forge}} - 1$. Since tx_{forge} has not yet been generated at round j by definition of r_{forge}, the honest miner's Input() at round j contains only $tx_{migrate}$ and no conflicting transaction. Therefore, this honest block only contains $tx_{migrate}$.

By the chain growth property, the chain grows by at least $\tau \cdot u = (1-\epsilon)f \cdot \lceil 4\lambda/(1-\epsilon)\rceil \geq 4\lambda f = 2k$ blocks during the u rounds after r_{migrate}. We distinguish two cases based on m.

- *Case $m \leq k$.* All m blocks generated during the migration window lie within the first $m \leq k$ positions of the $2k$ newly grown blocks. In particular, every such block, including the honest block containing $tx_{migrate}$, is at depth at least $2k - m \geq 2k - k = k$ from the end of C.
- *Case $m > k$.* Since $m > k \geq \lceil 2\lambda f\rceil$, the chain quality property guarantees that at least one honest block exists among the *first* k blocks of the m-block segment. This honest block was also generated during a round within $[r_{\mathrm{migrate}}, r_{\mathrm{forge}}-1]$, so it contains $tx_{migrate}$, and it lies at depth at least $2k-k = k$ from the end of C.

In both cases, $tx_{migrate}$ appears at depth more than k in C after u rounds. By the common prefix property, all honest parties share this block, and thus all honest parties report $tx_{migrate}$ at depth more than k. Therefore, migration liveness holds.

Next, we prove that if migration liveness holds, an honest block is generated during the migration window. We prove this by contradiction. Assume:

(1) Migration liveness holds: all honest parties report $tx_{migrate}$ at depth more than k from the end of C after u rounds.
(2) No honest block generated during migration window exists in C.

We derive a contradiction in each of the following cases.

- *Case $m \geq k$.* By the chain quality property, any $\lceil 2\lambda f\rceil$ consecutive blocks in an honest party's chain during a typical execution contain at least one honest block. Since $m \geq k = \lceil 2\lambda f\rceil$, the m blocks generated during the migration window must contain at least one honest block. This directly contradicts assumption (2).
- *Case $m < k$.* By assumption (1), $tx_{migrate}$ must be reported at depth more than k from the end of C. By the chain growth property, the chain grows by at least $2k$ blocks during the u rounds. For $tx_{migrate}$ to appear at depth more than k, there

must exist a block containing $tx_{migrate}$ in the k suffix of $C^{\lceil k}$. Since adversarial miners ignore $tx_{migrate}$ by the worst-case adversarial model, only an honest block can contain $tx_{migrate}$. Therefore, assumption (1) requires the existence of an honest block containing $tx_{migrate}$ in the k suffix of $C^{\lceil k}$. We now show that no such honest block can exist under the assumption (2). Consider blc_h the first honest block generated at round $j \geq r_{\text{forge}}$. This blc_h theoretically cannot contain $tx_{migrate}$. By the transaction consistency (Corollary 3.10), no chain can contain both $tx_{migrate}$ and tx_{forge}, since $C(tx_{migrate}, tx_{forge}) = 1$. Once tx_{forge} is included in any preceding adversarial block of blc_h, by this lemma, it cannot include $tx_{migrate}$. Even if the blc_h is generated at round r_{forge}, it cannot contain $tx_{migrate}$ because tx_{forge} is generated at the same round and both transactions are in Input() simultaneously, and the quantum adversary's prioritization assumption ensures that honest miners adopt tx_{forge} over $tx_{migrate}$. Therefore, we conclude that no honest block in C contains $tx_{migrate}$ under assumption (2). This contradicts assumption (1), which requires $tx_{migrate}$ to appear at depth more than k in C.

In both cases, assumptions (1) and (2) cannot simultaneously hold, completing the proof. □

4.3 The Tolerable Duration Threshold Δ^*

By lemma 4.2, if migration liveness holds, the blockchain within the PQC migration also exhibits the on-spend attack resistance against a quantum adversary $\mathcal{A}_q$ in a typical execution. Lemma 4.5 reduces migration liveness to a simple condition on the chain C of honest parties: the existence of at least one honest block in the migration window. Thus, the original question reduces to: under what condition is the honest block guaranteed to be within the migration window in a typical execution?

We now derive this condition of $\mathcal{A}_q$. Recall the definition of $r_{\text{forge}} = r_{\text{migrate}} + \Delta_{\text{eff}}$, therefore the number of blocks in the migration window m is determined by the quantum attacker's capability Δ_{eff}. Specifically, by the chain growth property, in a typical execution with $\Delta_{\text{eff}} \geq \lambda$ rounds, the chain grows by at least $\tau \cdot \Delta_{\text{eff}} = (1-\epsilon) f \cdot \Delta_{\text{eff}}$ blocks in the migration window. Therefore:

$$m \geq (1-\epsilon) f \cdot \Delta_{\text{eff}}.$$

By the chain quality property, in a typical execution, any $\lceil 2\lambda f \rceil$ consecutive blocks in an honest party's chain contain at least one honest block. Therefore, if $m \geq 2\lambda f$, the honest block is guaranteed to be in the migration window. To ensure $m \geq 2\lambda f$, it suffices to require:

$$(1-\epsilon) f \cdot \Delta_{\text{eff}} \geq 2\lambda f,$$

which simplifies to $\Delta_{\text{eff}} \geq 2\lambda/(1-\epsilon)$.

Definition 4.6 (Tolerable $\mathcal{A}_q$ Duration Threshold). Given an effective forgery delay Δ_{eff}, set

$$\lambda := \left\lfloor \frac{(1-\epsilon)\Delta_{\text{eff}}}{2} \right\rfloor,$$

so that $\Delta_{\text{eff}} \geq 2\lambda/(1-\epsilon)$ holds by construction. Definition 3.1 requires $\lambda \geq 2/f$, which holds precisely when

$$\Delta_{\text{eff}} \geq \Delta^* := \left\lceil \frac{4}{(1-\epsilon) f} \right\rceil.$$

Remark 4.7 (Operational reading). Since k and u are fixed by λ, the parametrization can be read in either direction. A deployed chain fixes λ, and hence k and u, from the security parameter κ, and Theorem 4.8 below then reads as a requirement $\Delta_{\text{eff}} \geq 2\lambda/(1-\epsilon)$ on the adversary. For the analysis it is more convenient to derive λ from Δ_{eff}, since the failure term becomes explicit in Δ_{eff}. This is well defined because Δ_q is a deterministic parameter of O_{shor} fixed by hardware (Definition 3.14), hence known at analysis time and independent of the execution's randomness.

4.4 Main Result

When $\Delta_{\text{eff}} \geq \Delta^*$, the honest block inclusion in the migration window is satisfied in a typical execution. Combining Lemma 4.2 and Lemma 4.5 with the tolerable $\mathcal{A}_q$ duration threshold Δ^*, we obtain the following.

Theorem 4.8 (On-spend Attack Resistance). *Fix Δ_{eff} and set $\lambda := \lfloor (1-\epsilon)\Delta_{\text{eff}}/2 \rfloor$, $k = \lceil 2\lambda f \rceil$, and $u = \lceil 4\lambda/(1-\epsilon) \rceil$. Under the Honest Majority Assumption, public transaction ledger protocol Π_{PL} satisfies the following:*

1. *If $\Delta_{\text{eff}} \geq \Delta^*$, then migration liveness holds, and hence on-spend attack resistance is maintained, with probability at least $1 - e^{-\Omega(\epsilon^2(1-\epsilon) f \Delta_{\text{eff}}/2)}$. In particular, the guarantee is overwhelming in κ whenever $f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$.*
2. *If $\Delta_{\text{eff}} < \Delta^*$, our sufficient condition no longer applies, and on-spend attack resistance is not guaranteed.*

This result formally characterizes the quantum computing capability to degrade blockchain security via an on-spend attack, parameterized by a simple intrinsic chain quantity: the honest mining success probability f. The threshold Δ^* can be interpreted as the sufficient condition for migration security in Phase 2. As long as the $\mathcal{A}_q$'s effective forgery delay $\Delta_{\text{eff}} \geq \Delta^*$, migration liveness holds with overwhelming probability once $f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$, and on-spend attacks are precluded. Once quantum computing capabilities advance to the point where Δ_{eff} falls below Δ^*, our analysis yields no statement, and unmigrated addresses cannot be certified as safe. Phase 2 should therefore be completed while Δ_{eff} remains large enough to meet this condition.

4.5 Generalized Threshold: Requiring Multiple Honest Blocks

The analysis above establishes that migration liveness holds when at least one honest block generated during the migration window is recorded in the chain. In practice, one may wish to require multiple honest blocks in this window, for several reasons:

- *Limited transaction space.* The number of migration transactions that fit in a single block may be strictly limited by the block size constraints.
- *Attacker blocking.* An attacker may attempt to block the inclusion of migration transactions by methods outside the

scope of our execution model, such as launching DDoS attacks against the mempool to cause congestion.

- *Hourglass model.* Proposals in the blockchain community, such as the hourglass spending rules [4], suggest limiting the number of ECDSA-sig transactions included per block during phase 2 in our PQC migration model.

We therefore generalize the analysis by requiring that $\ell_h \geq 1$ honest blocks be present in the migration window to ensure the $tx_{migrate}$ inclusion in the migration window. ℓ_h therefore means the migration robustness.

Definition 4.9 (ℓ_h-Migration Liveness). The ℓ_h-migration liveness property, parameterized by $\ell_h, u, k \in \mathbb{N}$, states the following: provided that tx_{migrate} is given as input to all honest parties continuously for u consecutive rounds starting from round r_{migrate}, then in the chain C of any honest party, there exist at least ℓ_h honest blocks generated after r_{migrate} to $r_{\text{forge}} - 1$, each appearing in more than k blocks from the end of C, and all honest parties report tx_{migrate} recorded in one of the ℓ_h honest blocks.

Deriving the generalized threshold. Under the substitution $\lambda := \lfloor (1-\epsilon)\,\Delta_{\text{eff}}/2 \rfloor$ of Definition 4.6, the baseline threshold $\Delta^* = \lceil 4/((1-\epsilon)f) \rceil$ is exactly the condition $\lambda \geq 2/f$ of Definition 3.1, and the migration window then contains at least $\lceil 2\lambda f \rceil$ blocks by construction, which is the minimum window length for the chain quality property to apply. When $\ell_h > 1$, two additional requirements arise: the window must contain at least ℓ_h honest blocks, and the depth k must be large enough to accommodate them.

Let m denote the number of blocks in C generated during the migration window. By the chain quality property, any $m \geq \lceil 2\lambda f \rceil$ consecutive blocks in an honest party's chain contain at least a fraction μ of honest blocks. Therefore, to guarantee ℓ_h honest blocks, we need $\mu \cdot m \geq \ell_h$, i.e., $m \geq \ell_h/\mu$. Since $m \geq (1-\epsilon) f \cdot \Delta_{\text{eff}}$ by chain growth, this gives

$$\Delta_{\text{eff}} \;\geq\; \frac{\ell_h}{\mu\,(1-\epsilon)\,f}\,.$$

For the second requirement, recall that $k = \lceil 2\lambda f \rceil$ bounds the depth in which the ℓ_h honest blocks of the migration window must reside, so the tail-bounds parameter must scale with ℓ_h as $\lambda \geq 2\ell_h/f$. Under the same substitution, this holds precisely when

$$\Delta_{\text{eff}} \;\geq\; \frac{4\,\ell_h}{(1-\epsilon)\,f}\,,$$

which generalizes the baseline Δ^* recovered at $\ell_h = 1$. Both requirements are linear in ℓ_h, and the chain quality requirement dominates if and only if $\mu < 1/4$, independently of ℓ_h. Combining both:

Definition 4.10 (Generalized Tolerable $\mathcal{A}_q$ Duration Threshold).

$$\Delta^*(\ell_h) \;=\; \max\left(\left\lceil \frac{4\,\ell_h}{(1-\epsilon)\,f} \right\rceil,\; \left\lceil \frac{\ell_h}{\mu\,(1-\epsilon)\,f} \right\rceil\right)$$

Theorem 4.11 (Generalized On-spend Attack Resistance). *Fix Δ_{eff} and set $\lambda := \lfloor (1-\epsilon)\Delta_{\text{eff}}/2 \rfloor$, $k = \lceil 2\lambda f \rceil$, and $u = \lceil 4\lambda/(1-\epsilon) \rceil$. Under the Honest Majority Assumption, the public transaction ledger protocol Π_{PL} satisfies the following:*

(1) *If $\Delta_{\text{eff}} \geq \Delta^*(\ell_h)$, then ℓ_h-migration liveness holds, and hence on-spend attack resistance is maintained, with probability at least $1 - e^{-\Omega(\epsilon^2(1-\epsilon)f\Delta_{\text{eff}}/2)}$, which is overwhelming in κ whenever $f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$.*

(2) *If $\Delta_{\text{eff}} < \Delta^*(\ell_h)$, the framework is not applicable, and on-spend attack resistance is not guaranteed.*

5 Implications for PQC Migration

5.1 Quantifying the Trade-off

The generalized threshold $\Delta^*(\ell_h)$ has f in the denominator of both terms. A larger f yields a smaller threshold and thus better quantum resistance, meaning that the chain can tolerate a stronger quantum adversary with shorter Δ_{eff}. However, the Honest Majority Assumption requires $3f + 3\epsilon < \delta$, so increasing f demands a larger honest majority advantage δ and limits the admissible adversarial share. We now quantify the trade-off found in the threshold.

Define the adversarial share $\alpha = t/n$. The Honest Majority Assumption is equivalent to

$$\alpha \;\leq\; \frac{1-\delta}{2-\delta}\,.$$

We evaluate the Honest Majority Assumption at its tightest boundary $\delta = 3f + 3\epsilon$, giving $f = (\delta - 3\epsilon)/3$. For the chain quality parameter, we use the exact bound from Theorem 3.8 evaluated at $t/(n-t) = 1-\delta$:

$$\mu \;=\; \frac{\delta}{2} + \frac{\delta^2}{2} - \frac{\epsilon}{1-\epsilon}\,.$$

Each value of f determines δ, μ, and hence $\Delta^*(\ell_h) = \max\{4\ell_h/((1-\epsilon)f), \ell_h/(\mu(1-\epsilon)f)\}$, as well as $\alpha_{\max} = (1-\delta)/(2-\delta)$. Figure 4 plots $\Delta^*(\ell_h)$ for $\ell_h \in \{1, 2, 5, 10\}$ and $\alpha_{\max}$ against f at $\epsilon = 0.05$.

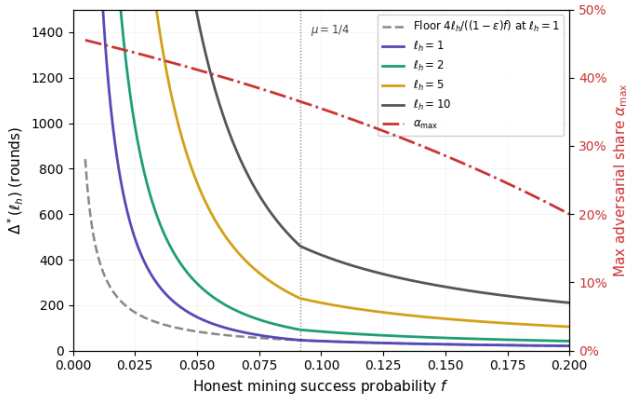


Figure 4: Relationship between honest mining success probability f and generalized tolerable $\mathcal{A}_q$ capability threshold $\Delta^*(\ell_h)$ with $\epsilon = 0.05$. $\Delta^*(\ell_h)$ in rounds (left axis), plotted against f for $\ell_h \in \{1, 2, 5, 10\}$. The gray dashed curve shows the baseline $4\ell_h/((1-\epsilon)f)$ at $\ell_h = 1$; each solid curve shows the second term $\ell_h/(\mu(1-\epsilon)f)$, which dominates $\Delta^*(\ell_h)$ when $\mu < 1/4$. The vertical line marks $\mu = 1/4$, common to all curves. The red dash-dotted curve (right axis) shows the maximum tolerable adversarial share $\alpha_{\max} = (1-\delta)/(2-\delta)$, evaluated at the tightest Honest Majority Assumption boundary $\delta = 3f + 3\epsilon$. As f increases, $\Delta^*(\ell_h)$ decreases while $\alpha_{\max}$ also decreases, visualizing the trade-off between quantum resistance and the honest majority requirement.

We observe trade-offs between parameters. First, $\alpha_{\max}$ decreases monotonically with f, and so does $\Delta^*(\ell_h)$. A chain that tolerates a faster $\mathcal{A}_q$ admits a smaller adversarial mining share. This extends the well-known f and honest-majority trade-off in the GKL framework to the quantum setting, yielding a trade-off between quantum resistance and the honest-majority requirement. Second, for fixed f, increasing ℓ_h raises $\Delta^*(\ell_h)$. A chain that requires more robust migration can only tolerate slower $\mathcal{A}_q$. This shows the trade-off between the blockchain's migration robustness and quantum resistance. Third, holding $\Delta^*(\ell_h)$ fixed while raising ℓ_h forces f upward, which in turn tightens $\alpha_{\max}$. Maintaining resistance against faster A_q while increasing migration robustness requires a smaller adversarial mining share. Therefore, the chain's quantum resistance, migration robustness, and adversarial-share margin cannot be improved independently.

5.2 Considerations for PQC Migration Practices

Chain-specific quantum resistance. The honest-mining success probability f is determined by each chain's protocol parameters, such as the block interval, mining difficulty, and network synchrony. As a result, different chains exhibit different values of f and thus different tolerable $\mathcal{A}_q$ duration threshold Δ^*. Our analysis, therefore, yields chain-specific quantum-resistance guarantees. Consider Bitcoin as a concrete example. GKL20 [16] estimates $0.02 \le f \le 0.03$ under the assumption that a full communication round takes up to 20 seconds. Substituting these values into $\Delta^* = \lceil 4/((1-\epsilon)f) \rceil$ with $\epsilon = 0.05$ gives $141 \le \Delta^* \le 211$ rounds, corresponding to an admissible forgery delay Δ_{eff} of roughly 47 to 70 minutes. We stress that this is a floor rather than a safety margin: it is the point below which the framework yields no statement, while an overwhelming guarantee additionally requires $f \cdot \Delta_{\text{eff}} = \Omega(\log^2 \kappa)$ and hence a substantially larger Δ_{eff}. Chains with shorter block intervals, such as Litecoin, which targets a 2.5-minute block interval compared to Bitcoin's 10 minutes, exhibit larger f and hence smaller Δ^* and admissible Δ_{eff}. Such chains theoretically maintain the on-spend attack resistance against faster quantum adversaries than Bitcoin does. However, a larger f also tightens the Honest Majority Assumption, lowering $\alpha_{\max}$. This means that $\mathcal{A}_q$ can corrupt fewer mining parties to violate the Honest Majority Assumption and, consequently, compromise the blockchain security assumptions, including migration liveness, at a lower economic cost. Thus, our observation extends the well-known trade-off between f and the honest-majority assumption to the quantum setting: a chain with a larger f may exhibit higher quantum resistance in terms of $\mathcal{A}_q$ execution time, but the financial cost of colluding with $\mathcal{A}_m$ to break the Honest Majority Assumption becomes lower.

Cost of migration robustness and key-exposure prevention. The robustness parameter ℓ_h can be made concrete by introducing chain-state quantities. Let N denote the total number of addresses that remain unmigrated at the start of Phase 2, and let d denote the per-block capacity for $tx_{migrate}$ inclusion being constrained by block size, transaction-fee competition, or explicit limits such as the hourglass spending rules [4]. In its simplest form, $\ell_h \approx N/d$. A tight characterization of ℓ_h requires modeling user issuance timing, fee bidding among users, and the distribution of assets across addresses, all of which involve game-theoretic analysis beyond the scope of this work.

This parameterization makes the practical cost of ℓ_h explicit. A large N entering Phase 2 directly or limiting d small inflates ℓ_h, which raises the tolerable $\mathcal{A}_q$ duration threshold $\Delta^*(\ell_h)$ and therefore lowers the chain's quantum resistance. This results in tightening the timeline for completing Phase 2. Restoring the threshold $\Delta^*(\ell_h)$ to a given target level requires increasing f, which in turn lowers $\alpha_{\max}$, as discussed above. In other words, a chain that enters Phase 2 with many unmigrated addresses or tightly limiting tx_{forge} inclusion in Phase 2 faces a forced trade-off between weaker quantum resistance and a tighter honest-majority requirement.

This observation has direct implications for planning blockchain PQC migration. For example, recent proposals [5, 12] aim to maintain at-rest resistance ($\Delta_{\text{head}} = 0$) for their addresses by eliminating on-chain exposure of verification keys or by rotating single-use ECDSA keys. While valuable in their own right, these strategies do not reduce N; if widely adopted in active PQC migration, they may even increase N by encouraging users to defer migration until Phase 2. A comprehensive migration plan must therefore complement key-exposure prevention with mechanisms that incentivize early migration to PQC-derived addresses during Phase 1, so that N at the start of Phase 2 stays small enough for the chain's f and $\alpha_{\max}$ to absorb the resulting ℓ_h.

5.3 Future Work

While our analysis establishes the first formal threshold for blockchain PQC migration security, several directions remain to sharpen the result.

Tighter thresholds via matching attack analysis. Our analysis establishes $\Delta^*(\ell_h)$ as a sufficient threshold: when $\Delta_{\text{eff}} \ge \Delta^*(\ell_h)$, ℓ_h-migration liveness holds with overwhelming probability. The converse direction, characterizing the precise condition in which on-spend attacks become feasible, remains open. Closing this gap requires analyzing the adversarial side of the threshold by identifying the tightest Δ_{eff} for which a concrete attack strategy succeeds. Such a matching lower-bound analysis would convert our sufficient threshold into a tight characterization, yielding a strict migration deadline rather than a conservative upper bound on tolerable quantum capability.

Tighter bounds via continuous-time analysis. Our analysis adopts the synchronous, discrete-round execution model of GKL15 [17] as the foundation, but we recognize that the resulting bounds are loose in this sense as well. The synchronous model was a deliberate starting point: it offers the cleanest setting for formalizing migration liveness and isolating the role of Δ_{eff}. Subsequent work by GKL15 and GKL20 has established tighter security bounds for the Bitcoin backbone against private mining attacks under continuous-time models that account for realistic network latency [11, 18]. Lifting our analysis into these frameworks would yield sharper thresholds and, correspondingly, more precise references for migration timelines.

Game-theoretic analysis. Our formalization establishes the relationship between the migration robustness ℓ_h and the threshold $\Delta^*(\ell_h)$, providing a theoretical basis for designing secure migration plans. The value of ℓ_h itself is determined by the migration process:

how many addresses must migrate, how the per-block capacity is allocated, and how users, $\mathcal{A}_q$, and miners interact through fee bidding. Turning our formal relationship into a practical implication for migration planning, therefore, requires a rigorous characterization of ℓ_h. As mentioned in Section 5.2, our $\ell_h \approx N/d$ is a first-cut characterization and remains loose; a tight bound requires game-theoretic modeling. Such a rigorous treatment of strategic interactions would yield a sharper bound on ℓ_h and, in turn, on $\Delta^*(\ell_h)$.

6 Conclusion

We presented the first formal security analysis of blockchain PQC migration against on-spend attacks. We defined migration liveness as the critical security property for migration and established a sufficient threshold for the tolerable quantum adversary capability, parameterized by the chain's honest mining success probability f and the migration robustness ℓ_h. Our analysis provides a theoretical basis for designing secure migration plans and quantifies the trade-offs among quantum tolerance, migration robustness, and the honest-majority requirement that any chain-wide migration plan must navigate.

7 Acknowledgments

This material is based upon work supported by the U.S. National Science Foundation under Cooperative Agreement No. 2113811. This work was conducted at the NSF Industry–University Cooperative Research Center Cyber SMART. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. National Science Foundation.

References

[1] Divesh Aggarwal, Gavin Brennen, Troy Lee, Miklos Santha, and Marco Tomamichel. 2018. Quantum Attacks on Bitcoin, and How to Protect Against Them. *Ledger* 3 (Oct. 2018). doi:10.5195/ledger.2018.127

[2] Ryan Babbush, Adam Zalcman, Craig Gidney, Michael Broughton, Tanuj Khattar, Hartmut Neven, Thiago Bergamaschi, Justin Drake, and Dan Boneh. 2026. Securing Elliptic Curve Cryptocurrencies against Quantum Vulnerabilities: Resource Estimates and Mitigations. arXiv:2603.28846 [quant-ph] https://arxiv.org/abs/2603.28846

[3] Foteini Baldimtsi, Konstantinos Chalkias, and Arnab Roy. 2025. Post-Quantum Readiness in EdDSA Chains. Cryptology ePrint Archive, Paper 2025/1368. https://eprint.iacr.org/2025/1368

[4] Hunter Beast and Michael Casey. [n. d.]. Hourglass spending rules. https://github.com/cryptoquick/bips/blob/hourglass/bip-hourglass.mediawiki Accessed: 2026-4-29.

[5] Hunter Beast and Ethan Heilman. [n. d.]. BIP360: Pay to Quantum Resistant Hash. https://github.com/cryptoquick/bips/blob/p2qrh/bip-0360.mediawiki Accessed: 2026-4-29.

[6] Bleap. [n. d.]. How Many Bitcoins Are Left to Mine in 2026? (Updated Halving Data, Supply Chart & Future Outlook). https://www.bleap.finance/en-us/blog/how-many-bitcoins-are-left-to-mine Accessed: 2026-03-13.

[7] Vitalik Buterin. 2014. A next-generation smart contract and decentralized application platform. https://ethereum.org/en/whitepaper/

[8] Madelyn Cain, Qian Xu, Robbie King, Lewis R. B. Picard, Harry Levine, Manuel Endres, John Preskill, Hsin-Yuan Huang, and Dolev Bluvstein. 2026. Shor's algorithm is possible with as few as 10,000 reconfigurable atomic qubits. arXiv:2603.28627 [quant-ph] https://arxiv.org/abs/2603.28627

[9] CoinLedger. [n. d.]. How Much Bitcoin is Lost Forever? [August 2025 Data]. https://coinledger.io/research/how-much-bitcoin-is-lost Accessed: 2026-4-29.

[10] Alexandru Cojocaru, Juan Garay, Aggelos Kiayias, Fang Song, and Petros Wallden. 2023. Quantum Multi-Solution Bernoulli Search with Applications to Bitcoin's Post-Quantum Security. *Quantum* 7 (March 2023), 944. doi:10.22331/q-2023-03-09-944

[11] Amir Dembo, Sreeram Kannan, Ertem Nusret Tas, David Tse, Pramod Viswanath, Xuechao Wang, and Ofer Zeitouni. 2020. Everything is a Race and Nakamoto Always Wins. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, USA) *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 859–878. doi:10.1145/3372297.3417290

[12] Ethresearch. [n. d.]. Achieving Quantum Safety Through Ephemeral Key Pairs and Account Abstraction. https://ethresear.ch/t/achieving-quantum-safety-through-ephemeral-key-pairs-and-account-abstraction/24273 Accessed: 2026-4-29.

[13] Ethresearch. [n. d.]. How to hard-fork to save most users' funds in a quantum emergency. https://ethresear.ch/t/how-to-hard-fork-to-save-most-users-funds-in-a-quantum-emergency/18901/1 Accessed: 2026-4-29.

[14] Ethresearch. [n. d.]. The road to Post-Quantum Ethereum transaction is paved with Account Abstraction (AA). https://ethresear.ch/t/the-road-to-post-quantum-ethereum-transaction-is-paved-with-account-abstraction-aa/21783 Accessed: 2026-4-29.

[15] Kigen Fukuda, Shin'ichiro Matsuo, Yuji Suga, and Tadahiko Ito. 2025. The Grand Challenge of PQC Migration: Analysis of Modern Blockchain and Intertwined Human Egoisms. Cryptology ePrint Archive, Paper 2025/1626. https://eprint.iacr.org/2025/1626

[16] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. 2014. The Bitcoin Backbone Protocol: Analysis and Applications. Cryptology ePrint Archive, Paper 2014/765. https://eprint.iacr.org/2014/765

[17] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. 2015. The Bitcoin Backbone Protocol: Analysis and Applications. In *Advances in Cryptology - EUROCRYPT 2015*, Elisabeth Oswald and Marc Fischlin (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 281–310.

[18] Peter Gaži, Aggelos Kiayias, and Alexander Russell. 2020. Tight Consistency Bounds for Bitcoin *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 819–838. doi:10.1145/3372297.3423365

[19] Ilias Giechaskiel, Cas Cremers, and Kasper Rasmussen. 2016. On Bitcoin Security in the Presence of Broken Crypto Primitives. Cryptology ePrint Archive, Paper 2016/167. https://eprint.iacr.org/2016/167

[20] Stephen Holmes and Liqun Chen. 2021. Assessment of Quantum Threat To Bitcoin and Derived Cryptocurrencies. Cryptology ePrint Archive, Paper 2021/967. https://eprint.iacr.org/2021/967

[21] Dragos I. Ilie, William J. Knottenbelt, and Iain D. Stewart. 2020. Committing to Quantum Resistance, Better: A Speed-and-Risk-Configurable Defence for Bitcoin Against a Fast Quantum Computing Attack. In *Mathematical Research for Blockchain Economy*, Panos Pardalos, Ilias Kotsireas, Yike Guo, and William Knottenbelt (Eds.). Springer International Publishing, Cham, 117–132.

[22] Joseph J. Kearney and Carlos A. Perez-Delgado. 2021. Vulnerability of blockchain technologies to quantum attacks. *Array* 10 (2021), 100065. doi:10.1016/j.array.2021.100065

[23] Ruslan Kysil, István András Seres, Péter Kutas, and Nándor Kelecsényi. 2025. poqeth: Efficient, post-quantum signature verification on Ethereum. Cryptology ePrint Archive, Paper 2025/091. doi:10.1145/3708821.3736193

[24] Howell Liu, Zhipeng Wang, and William Knottenbelt. 2025. Towards Building Post-Quantum Secure Ethereum. In *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 1–5. doi:10.1109/ICBC64466.2025.11114696

[25] Jameson Lopp, Christian Papathanasiou, Pierre-Luc Dallaire-Demers, Joe Ross, Ian Smith, and Steve Vaile. [n. d.]. Post Quantum Migration and Legacy Signature Sunset. https://arc.net/l/quote/fgeoxtpm Accessed: 2026-4-29.

[26] Long Meng and Liqun Chen. 2021. An Enhanced Long-term Blockchain Scheme Against Compromise of Cryptography. Cryptology ePrint Archive, Paper 2021/1606. https://eprint.iacr.org/2021/1606

[27] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system.

[28] Muhammad Taha Ramzan and Stelvio Cimato. 2025. Blockchain in the Quantum Era: Surveying Security Challenges and Post-Quantum Cryptography. In *2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC)*. 2218–2223. doi:10.1109/COMPSAC65507.2025.00311

[29] Masashi Sato and Shin'ichiro Matsuo. 2017. Long-Term Public Blockchain: Resilience against Compromise of Underlying Cryptography. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. 1–8. doi:10.1109/ICCCN.2017.8038516

A Generative AI Usage

Tools used. Claude (Anthropic) and Grammarly were used during the preparation of this manuscript.

Scope of use.

- **Writing assistance.** Claude was used throughout the paper to revise and polish the author-drafted text's English style. Grammarly was used for grammar and spelling corrections.

The technical content, claims, definitions, lemmas, theorems, and proofs were conceived, developed, and written by the authors.

- **Code for figure generation.** The Python (matplotlib) script used to generate Figure 4 was drafted with assistance from Claude. The underlying closed-form expressions, parameter values, axis labels, and the curves plotted were specified by the authors.

Validation. The authors critically reviewed all AI-assisted text and code, verifying that they accurately reflect the intended technical content.