QuBit One: A Peer-to-Peer Electronic Cash System Secured Against Quantum Computers

Aurel Thornquist

Nostr: npub1xd0ay3faps89aj68ytyzem3s0hud4zhavy5zphzagx2ruzm823ssawryau

Version 1.0

Abstract

Bitcoin showed that a purely peer-to-peer electronic cash system can work without a trusted third party. Its proof-of-work and its addresses rest on hash functions, which a quantum computer weakens only modestly. Its signatures rest on elliptic curves, which a large quantum computer would break outright. We propose a new chain, QuBit One (QBO), that keeps Bitcoin's design, code base and monetary policy, and replaces every signature with standardized post-quantum schemes. Each output commits to two keys: an ML-DSA key for everyday spending and a hash-based SLH-DSA key kept hidden as a fallback, so that the network can retire the first scheme by soft fork without anyone having to move coins in a hurry. Outputs that a quantum computer could steal are forbidden by consensus from the first block. Difficulty is adjusted at every block so that a new chain cannot be mined out in hours by whoever arrives first, and there is no premine, no founder reserve and no sale. After an initial window reserved for independent miners, the chain is merge-mined with Bitcoin so that it inherits the security of the largest SHA-256 network.

1. Introduction

A Bitcoin coin is a chain of digital signatures [1]. The owner transfers it by signing a hash of the previous transaction and the public key of the next owner. The whole system therefore depends on one assumption: that nobody can forge a signature without the private key.

Today that assumption rests on the elliptic curve discrete logarithm problem. Shor's algorithm [4] solves that problem in polynomial time on a sufficiently large fault-tolerant quantum computer. Such a machine does not exist yet, and serious estimates of when it might range from several years to several decades. When it arrives, any coin whose public key has been revealed can be spent by the attacker. That includes early pay-to-public-key outputs, every reused address, every Taproot output, and every coin in the window between the moment its spending transaction is broadcast and the moment it is confirmed.

Bitcoin can and probably will migrate to post-quantum signatures. A migration of a running system is slow, however: it needs a soft fork, new wallets, and every holder must move their coins to new outputs before the attacker appears. Coins whose owners have lost their keys, or are no longer alive, can never move.

We take the opposite approach. Instead of migrating an existing chain, we start a new one in which no coin ever depends on elliptic curves. The rest of Bitcoin is kept as it is, because its rules have been tested in an adversarial environment for over fifteen years. QBO is a fork of Bitcoin Core v32.0 [10]: the first

commit of its repository is the unmodified upstream code, and the second holds every QBO change, so that the whole difference can be reviewed as a single diff.

2. Threat Model

We assume an adversary who will, at an unknown future date, own a large fault-tolerant quantum computer, and who records the blockchain until then. We assume that hash functions remain secure up to the generic quantum speedup.

Primitive	Used for	Best known quantum attack	Effect
ECDSA, Schnorr (secp256k1)	Bitcoin signatures	Shor: polynomial time	Broken
SHA-256	Proof-of-work, Merkle trees, commitments	Grover: square-root speedup	256-bit preimage resistance falls to about 128 bits
ML-DSA-44 (FIPS 204) [2]	QBO everyday signatures	None better than classical	Module lattices, NIST security category 2
SLH-DSA-SHA2-128s (FIPS 205) [3]	QBO fallback signatures	Grover only	Security reduces to SHA-256

Grover's algorithm [5] does not give a practical advantage in mining. Its speedup applies to a single long sequential search, it cannot be parallelized efficiently, and each step must run on error-corrected qubits that are many orders of magnitude slower than a SHA-256 ASIC. Proof-of-work is therefore kept unchanged.

The design rule that follows is simple: on QBO, nothing that can be stolen by a quantum computer is ever written to the chain.

3. Transactions

QBO keeps Bitcoin's unspent transaction output model, its script system, its segregated witness [6] and its transaction format. One consensus rule is added, called `quantumsafe`. A transaction is valid only if each of its outputs is one of the following:

1. a witness version 2 output with a 32-byte program, described in Section 4;
2. a witness version 3 to 16 output, reserved for future soft forks;
3. an `OP_RETURN` output, which carries data and can never be spent.

Every other output type is rejected, both in blocks and in the memory pool: pay-to-public-key, pay-to-public-key-hash, multisig, pay-to-script-hash, SegWit v0 and Taproot. A user cannot lock coins to an elliptic curve key, even by mistake. The rule is a buried deployment, active from the genesis block on every network.

The same rule also makes invalid any transaction whose serialization without witness is exactly 64 bytes. Such a transaction has the size of an inner node of the Merkle tree, and could be used to fool a light client into accepting a fake transaction as part of a block. Bitcoin has carried this ambiguity since 2009, and a soft fork proposed for it (BIP 54 [12]) contains the same rule. On QBO the rule costs nothing: a witness version 2 output alone takes 43 bytes, so a 64-byte transaction could never pay anyone, and could only carry a tiny `OP_RETURN`.

All of Bitcoin's historical soft forks (P2SH, strict DER, CHECKLOCKTIMEVERIFY, CHECKSEQUENCEVERIFY, SegWit, Taproot) are active from block 1. Transaction-level time locks (`nLockTime`) and relative time locks (BIP 68 sequence numbers) apply to QBO transactions. Script-level features do not: a QBO output is a single owner's two-lock key tree, with no script. Multi-signature wallets, hash time-locked contracts and `OP_CHECKLOCKTIMEVERIFY`-style conditions require a future soft fork that adds post-quantum scripts (see Section 10).

4. Two-Key Outputs

A post-quantum scheme built on young mathematics might one day be broken by classical cryptanalysis. Lattice-based signatures were standardized in 2024 and are well studied, but they have nowhere near the history of hash functions. Hash-based signatures rest on the strongest assumption available, the security of SHA-256 itself, but are about three times larger.

QBO uses both. Each output commits to two public keys in a Merkle tree of two leaves, in the manner of Taproot [7]:

```
leaf_mldsa  = TaggedHash("QBO/leaf", 0x01 || mldsa_pubkey)
leaf_slhdsa = TaggedHash("QBO/leaf", 0x02 || slhdsa_pubkey)
program     = TaggedHash("QBO/branch", sort(leaf_mldsa, leaf_slhdsa))
```

The output script is `OP_2 <program>`. Only the 32-byte hash is stored in the output, so neither public key is revealed before the coin is spent.

To spend, the owner reveals one leaf. The witness has exactly four elements:

```
[ path (1 byte), public key, signature, hash of the other leaf (32 bytes) ]
```

The verifier recomputes the leaf from the path and the public key, combines it with the supplied sibling hash, checks that the result equals the program, and verifies the signature with the scheme the path names. Path 1 is ML-DSA-44 and path 2 is SLH-DSA-SHA2-128s; any other path is invalid.

Scheme	Public key	Signature	Witness for one input
ML-DSA-44 (path 1)	1,312 bytes	2,420 bytes	about 3,775 bytes
SLH-DSA-SHA2-128s (path 2)	32 bytes	7,856 bytes	about 7,930 bytes

In normal use only path 1 is spent, and the SLH-DSA key stays hidden behind its hash. A house has a front door and a trapdoor whose key only the owner holds. If ML-DSA is ever broken, a soft fork can make path 1 invalid. From that block on, owners spend through path 2, without having had to move their coins in advance, and the attacker gains nothing from public keys revealed in the past.

Stateful hash-based schemes such as XMSS and LMS were rejected: they require the signer to never reuse a counter, and a single backup restored from an old state would allow forgeries. FN-DSA (Falcon) has smaller signatures but is harder to implement without side channels, and its standard was not final when this design was fixed.

5. Signature Message

Both paths sign the same 32-byte message, computed as a tagged SHA-256 hash of:

- the transaction version and lock time;
- the hashes of all inputs' previous outpoints, of all spent amounts, of all spent scripts and of all input sequence numbers;
- the hash of all outputs;
- the index of the input being signed.

There is a single signature mode, which commits to the whole transaction. Committing to every spent amount and script lets an offline signer, such as a hardware wallet, know exactly what it is signing without seeing the previous transactions.

Signatures are produced with the FIPS 204 and FIPS 205 context string `"QBO/v1/"` followed by the network name (`"QBO/v1/main"`, `"QBO/v1/testnet4"`, `"QBO/v1/regtest"`). A signature made for QBO cannot be replayed in another protocol that uses the same keys, nor on another QBO network, and a future signature version can use a new context.

6. Proof-of-Work and Difficulty

Proof-of-work is Bitcoin's: double SHA-256 over an 80-byte header, with a target block interval of 10 minutes. Existing SHA-256 hardware, from a small home miner to an industrial ASIC, can mine QBO without change.

The difficulty rule is different. Bitcoin retargets every 2,016 blocks and changes difficulty by at most a factor of four each time. On a new chain that starts at minimum difficulty, this rule is a gift to whoever arrives first with modern hardware. A single 50 TH/s machine starting at difficulty 1 would find the first 2,016 blocks in under a second, and would need twelve retarget periods to reach the right difficulty: about 24,000 blocks, or 1.2 million coins, in roughly a quarter of an hour. That would be a premine in all but name.

QBO instead adjusts difficulty at every block with the absolutely scheduled exponential rule aserti3-2d [8], in service on another SHA-256 chain since November 2020. The target of the next block is

$$T_{n+1} = T_{\text{anchor}} \cdot 2^{(t_n - t_{\text{genesis}} - 600\,n)/7200}$$

where t_n is the timestamp of the last block, n its height, and T_{anchor} the starting target. Each time the chain gets two hours ahead of its schedule, difficulty doubles; each time it falls two hours behind, difficulty halves. The rule works like a thermostat that corrects at every block instead of once every two weeks. It uses integer arithmetic only, and the anchor is the genesis block, so every node computes the same target from the same headers.

The starting difficulty on the main network is 2^4 (compact target `0x1c0ffff0`). That corresponds to about 115 MH/s for one block every 10 minutes, the hash rate of one ordinary desktop computer. The chain is meant to be started by ordinary computers, and the rule raises the difficulty within hours when specialized hardware arrives.

Three safeguards complete the rule. During the first 2,016 blocks, the difficulty never falls below its starting value, so a genesis block whose timestamp lies in the past cannot be used to mine a burst of easy blocks. If no block is found for more than two hours, for example because a large miner leaves, each further full 10 minutes in a block's timestamp lowers its difficulty by a factor of $2^{1/12}$, so the chain cannot freeze. Finally, a block may be dated at most 10 minutes ahead of a node's clock instead of Bitcoin's two hours, which limits how far a miner can push the schedule with false timestamps.

7. Issuance and Fair Launch

The monetary policy is Bitcoin's. The first blocks pay 50 QBO, the reward halves every 210,000 blocks, and the total supply converges to 21 million QBO. One QBO is divided into 100,000,000 photons, the smallest unit, as one bitcoin is divided into satoshis. A photon is a quantum of light: it cannot be split.

The launch follows these rules:

- **No premine.** The only output of the genesis block is an `OP_RETURN`, so the first reward belongs to nobody.
- **Proof of a fresh start.** The genesis block is mined on launch day. Its message contains a newspaper headline of that day and the height of the latest Bitcoin block at that time, which proves that no QBO block could have been mined before. Its `OP_RETURN` output also carries the author's Nostr public key, so that announcements signed with that key can be tied to the chain itself.
- **No reserve, no sale, no administrator.** There is no founder allocation, no foundation holding coins, no token sale and no key that can change the rules or freeze funds.
- **Equal start.** The source code, the genesis block and the launch announcement are published at the same moment. The founder's computers mine from that moment, under the same rules as everybody else and never before, and the first blocks they find are announced as such.
- **Verifiable software.** Every release is identified by the full hash of its source commit and by the SHA-256 hashes of its packages, published with a signature from the author's key. Fully reproducible builds are planned, so that anyone can check that a download matches the published source.

8. Merged Mining

A new SHA-256 chain that stands alone is fragile: a small fraction of Bitcoin's hash power could rewrite it. Merged mining lets a Bitcoin miner work on both chains at once [9]. The QBO block hash is committed in the miner's Bitcoin coinbase transaction, and a QBO block is accepted if the Bitcoin header that carries this commitment meets the QBO target. The same work then secures both chains at no extra cost to the miner, and the QBO header remains easy to verify.

Merged mining also changes who can mine profitably. Once large Bitcoin pools merge-mine QBO, independent miners will hold a much smaller share of its rewards. For this reason merged mining is planned to activate only at block 21,000, about 146 days after launch, and not before. Until then, QBO is mined only by miners who choose it, and the per-block difficulty rule of Section 6 keeps rewards proportional to the hash power each one brings.

This window has a cost that we state plainly: during the first 21,000 blocks, the chain is only as secure as its own hash rate. Exchanges and merchants should require more confirmations in that period.

Merged mining is not in the first release. Because it makes valid some blocks that release rejects, it cannot be a soft fork: it will be the one planned exception to the rule of Section 10, a coordinated upgrade. Its exact specification and code will be published, reviewed and tested on a public test network, and released at least two months before the activation height, so that every node can upgrade in time. A node that does not upgrade would stop following the chain at that height. If the upgrade is not ready, or not widely adopted, the height moves later; it never moves earlier.

9. Space and Fees

Verifying an ML-DSA-44 signature costs a computation of the same order as verifying an ECDSA signature. The real cost of post-quantum signatures is space. A transaction with one input and two outputs weighs about 141 virtual bytes in Bitcoin and about 1,081 virtual bytes in QBO, roughly 7.7 times more. A rare spend through the SLH-DSA path weighs about 2,120 virtual bytes. Mailing a parcel instead of a postcard does not slow the mailman down, but his bag fills up faster.

QBO keeps Bitcoin's block weight limit of 4 million weight units, so that a full node can still run on inexpensive hardware. That gives about 900 simple transactions per block, about 1.5 per second. Measured on a test network: 962 one-input, one-output transactions fill a block of 3.7 MB. If every block were full, the chain would grow by about 195 GB per year, against about 100 GB for Bitcoin today. Four measures limit the impact:

1. **Small unspent outputs.** Each output stores only a 32-byte hash. Keys and signatures appear only in the witness of the spending transaction, so the set of unspent outputs that every node keeps in memory grows no faster than Bitcoin's.
2. **Witness discount.** As in SegWit, witness bytes count for one quarter of base bytes.
3. **Pruning.** As in Bitcoin, a node may discard old blocks once validated and keep only the headers, the recent blocks and the unspent output set.

4. **Off-chain payments, later.** Payment channels could move most everyday payments off-chain, but they need multi-signature and hash time-locked outputs, which QBO does not have yet. Until a soft fork adds post-quantum scripts, every payment is an on-chain transaction.

The block weight limit can be raised later if needed, but only by broad agreement of the network.

10. Upgrades

No code can prevent a fork. Anyone can copy the software and change a rule, and the value of a chain comes from the people who choose to run it. What the design can do is make the legitimate path to change obvious and safe.

- **Soft forks by default.** New rules only tighten existing ones, so older nodes keep following the chain. The one planned exception is merged mining (Section 8), a coordinated upgrade announced months in advance.
- **Reserved versions.** Witness versions 3 to 16 are reserved for future signature schemes and for post-quantum scripts (multi-signature, hash time locks, the building blocks of payment channels), which can be added without touching existing outputs.
- **A fallback in every output.** The SLH-DSA path of Section 4 lets the network survive the failure of ML-DSA.
- **Few fixed rules.** The 21 million limit, the 10-minute interval and the absence of any administrative key are not meant to change.

11. Distribution and Network Privacy

The source code is hosted on Radicle, a peer-to-peer code collaboration network: the repository is identified by a content-addressed ID and replicated by independent seed nodes, so no single company can take it down, and anyone can read or clone it without an account. Announcements are made on Nostr and signed with the author's key.

The node can run in a mode where every connection goes through Tor: it joins peers by their `.onion` addresses only, accepts connections on its own `.onion` address, and listens on no public IP address. The packaged miner offers the same mode and refuses to start if Tor does not answer, so that a misconfiguration fails safely instead of connecting in the clear. Miners who prefer ordinary connections can of course use them.

12. Privacy

Privacy works as in Bitcoin: transactions are public, and owners are known only by their keys. Wallets use a new key pair for each address by default. An output reveals nothing about its owner's keys before it is spent, and when it is spent through path 1, the SLH-DSA key remains hidden behind its hash.

13. Simplified Payment Verification

Block headers are unchanged: 80 bytes, about 4.2 MB per year. A light client can verify that a transaction is in the chain by downloading the headers and a Merkle branch, exactly as described in [1]. Phones are well suited to be light wallets. They are not suited to mining, which belongs to dedicated SHA-256 hardware.

14. Calculations

We simulate the difficulty rule of Section 6 when a hash rate H arrives all at once on a chain starting at difficulty 2^4, with each block taking its expected time. We count the blocks until the interval is back within 10% of 10 minutes.

Hash rate arriving	Blocks until about 10 minutes	Time elapsed
140 MH/s (one desktop computer)	13	1.9 hours
1.1 TH/s (one small home miner)	197	6.7 hours
50 TH/s	263	6.6 hours
500 TH/s	303	6.7 hours

A miner who brings four million times the expected hash rate gains about 300 blocks, 15,000 QBO or 0.07% of the supply, and the extra blocks fall within the first hours. With the 2,016-block rule, the same miner would have taken tens of thousands of blocks before the difficulty caught up.

The probability that an attacker with a share q of the hash power catches up from z blocks behind is the one given in [1]: it falls exponentially with z while $q < 0.5$. Merged mining raises the hash power an attacker must gather from QBO's own hash rate to that of the merge-mining Bitcoin miners.

15. Status and Limitations

QBO is new and experimental software. The node passes Bitcoin Core's unit and functional tests adapted to its rules, signature test vectors produced by an independent implementation, and an end-to-end test that pays with ML-DSA, spends through SLH-DSA, and checks that altered signatures, wrong keys, unknown paths, old address types and 64-byte transactions are rejected. Several pieces are still missing:

- a full wallet that manages post-quantum keys (a minimal tool exists);
- the merged mining implementation of Section 8;
- a signature cache and policy limits on the verification cost of witness version 2 spends;
- an independent security audit. The code has been through repeated internal adversarial reviews, but no outside firm has audited it, and early users should treat it as experimental software. Reviews are welcome; the repository explains how to report a flaw privately.

The schemes themselves carry a risk. ML-DSA and SLH-DSA are standards, but implementations are young, and QBO uses the liboqs library [11] for both. Transactions are larger than in Bitcoin, so fees per

payment will be higher at equal demand for block space. In its first months the chain will be secured by a small hash rate.

16. Conclusion

We have proposed a system for electronic cash that keeps Bitcoin’s rules and replaces the only part of it that a quantum computer would break. Each output commits to a lattice-based key for daily use and to a hash-based key held in reserve, and outputs that could be stolen by a quantum computer cannot be created. Difficulty follows the hash rate at every block, so a new chain can start fairly, and merged mining later lets it share the security of the largest proof-of-work network. There is no premine and no one in charge. Nodes vote with their hash power, and the rules hold for as long as the people who run them choose to keep them.

References

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008.

[2] NIST, “FIPS 204: Module-Lattice-Based Digital Signature Standard,” August 2024.

[3] NIST, “FIPS 205: Stateless Hash-Based Digital Signature Standard,” August 2024.

[4] P. W. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 1994.

[5] L. K. Grover, “A fast quantum mechanical algorithm for database search,” Proceedings of the 28th Annual ACM Symposium on Theory of Computing, 1996.

[6] E. Lombrozo, J. Lau, P. Wuille, “BIP 141: Segregated Witness (Consensus layer),” 2015.

[7] P. Wuille, J. Nick, A. Towns, “BIP 341: Taproot: SegWit version 1 spending rules,” 2020.

[8] J. Toomim, M. Lundeberg, “aserti3-2d: absolutely scheduled exponentially rising targets, integer version,” 2020.

[9] Namecoin, “Merged mining specification,” 2011.

[10] Bitcoin Core, version 32.0, https://github.com/bitcoin/bitcoin.

[11] Open Quantum Safe, “liboqs,” https://openquantumsafe.org.

[12] A. Poinsot, “BIP 54: Consensus Cleanup,” 2025.