QuantumHarmony Light Paper

Version 1.5 — January 2025
DOI: Pending Zenodo Publication

Sylvain Cormier
QuantumVerse Protocols
Paraxiom Research

Changelog v1.5: Added Web Interface section. Added Research Publications with DOIs (6 papers on Zenodo). Expanded references.

v1.4: Added Quantum P2P networking section (ML-KEM-1024, Falcon-1024, QKD hardware interface). Expanded Proof of Coherence with P2P integration details.

v1.3: Added Use Cases section (QCAD, Fideicommis, Pedersen). Expanded governance documentation. Added Triple Ratchet encryption and 512-segment toroidal mesh documentation.

v1.2: Added MEV protection documentation. Corrected finality description—QuantumHarmony provides deterministic BFT finality via the Coherence Gadget, not probabilistic finality.

Abstract

QuantumHarmony is a Layer 1 blockchain built on Substrate that replaces quantum-vulnerable cryptographic components with post-quantum alternatives. This document describes what the system does, how it works, and its current state.

1 Problem Statement

1.1 Quantum Computing Threat

Current blockchains rely on cryptographic primitives that quantum computers can break:

Primitive	Algorithm	Quantum Attack
Signatures	ECDSA, Ed25519	Shor's Algorithm
Finality	BLS (GRANDPA)	Shor's Algorithm
Hashing	Blake2b	Grover's Algorithm

Fact: NIST estimates cryptographically relevant quantum computers could exist within 10–15 years. Blockchain addresses and signed transactions recorded today become vulnerable once such computers exist.

1.2 What QuantumHarmony Changes

Component	Standard Substrate	QuantumHarmony
Signatures	Ed25519 / ECDSA	SPHINCS+ (NIST PQC)
Block Hashing	Blake2b	Keccak-256 (SHA-3)
Finality Gadget	GRANDPA (BLS)	Coherence Gadget (Falcon1024)
Randomness	VRF	Quantum-enhanced VRF (optional QKD)

2 Technical Implementation

2.1 SPHINCS+ Signatures

SPHINCS+ is a stateless hash-based signature scheme standardized by NIST in 2024. Its security relies solely on hash function properties, not discrete logarithms or elliptic curves.

Trade-offs:

- Signature size: approximately 8–50 KB
- Slower signing compared to Ed25519
- Verification time comparable to classical schemes

Implementation is provided via `pallet-sphincs-keystore`.

2.2 Keccak-256 Hashing

Keccak-256 (SHA-3) replaces Blake2 throughout the runtime.

- 256-bit output provides 128-bit post-Grover security
- 1600-bit sponge state
- Standardized and widely audited

2.3 Triple Ratchet Encryption

Validator-to-validator communication uses a **Triple Ratchet** protocol combining three key rotation mechanisms:

1. **Falcon Ratchet**: Long-term post-quantum signatures (slow rotation)
2. **Merkle Ratchet**: Hierarchical key derivation (periodic rotation)
3. **Symmetric Ratchet**: Ephemeral session keys (per-message rotation)

This provides forward secrecy: compromise of current keys does not reveal past messages.

2.4 512-Segment Toroidal Mesh

Runtime execution is parallelized across an 8×8×8 toroidal mesh (512 segments):

- Each segment handles a subset of accounts
- Parallel transaction execution within segments
- Cross-segment communication via 6 neighbors (3D torus)
- Load balancing with automatic rebalancing
- Maximum 3 hops between any two segments

Implementation: `pallet-runtime-segmentation`

2.5 Quantum-Secured P2P Networking

Validator-to-validator communication uses a fully post-quantum secured P2P layer:

Identity & Key Exchange:

- **Falcon-1024**: Node identity and message signing (NIST PQC)
- **ML-KEM-1024** (Kyber): Key encapsulation for session establishment (NIST PQC)
- **AES-256-GCM**: Authenticated encryption for message confidentiality

Protocol flow:

1. Node generates Falcon-1024 signing keypair + ML-KEM-1024 KEM keypair
2. Session initiation: ML-KEM encapsulation creates shared secret
3. Shared secret derives AES-256 session key
4. All messages signed with Falcon-1024, encrypted with AES-256-GCM
5. Automatic key rotation (default: 1 hour)

QKD Hardware Integration: When QKD hardware is available, session keys can be derived from QKD-generated entropy instead of ML-KEM. Supported vendors (stubs ready): Toshiba, ID Quantique, QuantumCTek, SK Telecom, NTT. Interface follows ETSI GS QKD 014.

2.6 Consensus and Finality

Block production: Aura (Authority Round).

Finality: Deterministic BFT finality via the **Coherence Gadget**, a post-quantum replacement for GRANDPA.

2.7 Coherence Gadget

The Coherence Gadget provides GRANDPA-equivalent deterministic finality:

GRANDPA	**Coherence Gadget**
BLS signatures	Falcon1024 signatures
Prevote / Precommit	STARK verification + coherence scoring
2/3 supermajority	2/3 supermajority
Finality proof	Finality Certificate

Protocol flow:

1. New block produced by Aura
2. Proof collection (entropy / coherence inputs)
3. Proof verification and scoring
4. Falcon1024 signing
5. Vote broadcast (encrypted)
6. Supermajority aggregation
7. Finality certificate generation

2.8 Proof of Coherence (PoC)

PoC is the consensus mechanism that combines quantum entropy with BFT finality.

With quantum hardware:

- QRNG / QKD entropy sources (Toshiba, Crypto4A, IdQuantique)
- STARK proofs verified with Winterfell
- QBER-based coherence scoring (threshold: 11%)
- QKD-derived session keys for validator P2P

Without hardware (fallback):

- Mock entropy sources for testing
- ML-KEM-1024 session keys for validator P2P
- Full BFT execution preserved
- Falcon1024 signatures provide post-quantum security
- Deterministic finality still guaranteed

Integration with P2P Layer:

- Coherence votes broadcast via quantum-secured P2P channels
- Vote encryption uses QKD-derived keys when available, ML-KEM otherwise
- Triple Ratchet provides forward secrecy for vote messages

Quantum hardware improves entropy quality but is not required for correctness or finality.

2.9 MEV Protection

QuantumHarmony provides native Maximal Extractable Value (MEV) protection at the protocol level.

The problem: In traditional blockchains, validators can reorder transactions (frontrunning), insert their own transactions (sandwich attacks), or censor specific transactions.

Solution:

1. Leader elected via quantum-seeded VRF (unpredictable)
2. Leader maintains qVRF-ordered priority queue
3. Leader compares priority queue against public mempool
4. Discrepant transactions are deleted

Reporter requirements: Every report must include a randomly generated nonce:

$$\texttt{tx\_hash} = \texttt{Hash}(\texttt{payload} \| \texttt{random\_nonce})$$

This ensures unique transaction hashes, prevents replay attacks, and enforces deterministic ordering.

Attack	**Mitigation**
Frontrunning	qVRF ordering is unpredictable
Sandwich attacks	Discrepancy detection removes injected txs
Transaction censorship	Leader rotation
Replay attacks	Random nonce per report

3 Use Cases

3.1 QCAD Stablecoin

Canadian dollar stablecoin (`pallet-stablecoin`):

- 1:1 CAD peg via oracle price feeds
- Collateralized vaults (150% minimum ratio)
- Liquidation engine for undercollateralized positions
- Stability fees paid in native token

3.2 Fideicommis Trusts

Quebec Civil Code compatible trust administration (`pallet-fideicommis`):

- Trust creation with grantor, trustee, beneficiaries
- Asset registration (on-chain and off-chain references)
- Distribution rules: time-locked, conditional, discretionary
- Trustee succession with multi-sig handoff

3.3 Pedersen Commitments

Zero-knowledge proofs on BLS12-381 (`pallet-pedersen-commitment`):

- Commit-reveal for MEV protection
- Range proofs for private amounts
- Binding + hiding properties

4 Governance System

QuantumHarmony includes standard Substrate governance pallets:

- Democracy
- Collective
- Treasury
- Scheduler

4.1 Academic Vouching

Credential verification system (`pallet-academic-vouch`):

- Institution registration (universities, certification bodies)
- Credential issuance with expiry
- On-chain voting for academic registration
- Vouch threshold for program acceptance

4.2 Ricardian Contracts

Human + machine readable legal contracts (`pallet-ricardian-contracts`):

- Dual format: legal prose + executable code
- Multi-party signing workflow
- Amendment tracking with version history
- State transitions: Draft → Active → Executed/Terminated

4.3 Notarial Services

Document attestation system (`pallet-notarial`):

- Hash attestation with timestamp proof
- Witness certification system
- Certificate generation
- Revocation with reason codes

5 Current State

Testnet: Operational (3 validators)
Block time: 6 seconds
Consensus: Aura + Coherence Gadget

What works:

- Block production with Aura + SPHINCS+
- Deterministic BFT finality via Coherence Gadget
- All governance and legal pallets
- STARK proof verification path
- Docker deployment

In progress:

- Production QKD hardware integration
- Multi-region validator expansion

Not done:

- Security audit
- Mainnet launch

6 Limitations

- Large post-quantum signatures (29 KB for SPHINCS+, 1.3 KB for Falcon1024)
- No BLS-style signature aggregation
- Non-standard Substrate tooling compatibility

7 Comparison

	QuantumHarmony	**Substrate**	**QRL**
Signatures	SPHINCS+ / Falcon	Ed25519 / BLS	XMSS
Finality	Deterministic BFT	GRANDPA (BFT)	PoW
MEV Protection	Native (qVRF)	No	No
Quantum HW	Optional	No	No

8 Web Interface

A web-based notarial interface is available for end users:

- Document attestation with SPHINCS+ signatures
- Contract creation and multi-party signing
- QCAD stablecoin transactions
- Fideicommis trust management
- Account creation with local key storage

Technical stack: SHA-256 document hashing, SPHINCS+-256s post-quantum signatures, real-time blockchain connection.

9 Research Publications

Theoretical foundations published on Zenodo with DOIs:

Paper	**Topic**	**DOI**
ERLHS	Hamiltonian framework for coherence-preserving ML	10.5281/zenodo.17928909
Karmonic Mesh	O(N log N) spectral consensus on toroidal manifolds	10.5281/zenodo.17928991
Proof of Coherence	QKD-based distributed consensus	10.5281/zenodo.17929054
Toroidal Mesh	10K TPS with SPHINCS+ via parallel verification	10.5281/zenodo.17931222
Toroidal Governance	Tonnetz manifold governance	10.5281/zenodo.17929091
Augmented Democracy	Coherence-constrained democratic infrastructure	Preprint

10 References

1. NIST Post-Quantum Cryptography Standardization (2024)
2. NIST FIPS 205: SPHINCS+
3. NIST FIPS 206: Falcon
4. NIST FIPS 203: ML-KEM
5. Substrate Developer Documentation
6. ETSI GS QKD 014: QKD Key Delivery API
7. Grover, L. "A Fast Quantum Mechanical Algorithm for Database Search" (1996)
8. Shor, P. "Algorithms for Quantum Computation" (1994)

Contact

Project: QuantumHarmony (QuantumVerse Protocols)
Technical Lead: Sylvain Cormier
Repository: `https://github.com/QuantumVerseProtocols/quantumharmony`

This document describes the system as implemented. No forward-looking claims are made.