BLEEP

Quantum Trust Network

Proven Execution. Quantum Foundation. Intent Native.

A Post-Quantum Cryptographic Foundation for Verifiable Decentralized Execution

WHITEPAPER — Protocol Version 5

Muhammad Attahir · May 2026

This document is provided for informational purposes only. It does not constitute financial advice, investment advice, or an offer to sell securities or digital assets. All protocol parameters and source references correspond to Protocol Version 5.

Abstract

Existing decentralized systems derive their security from cryptographic assumptions — integer factorization hardness, discrete logarithm intractability, and elliptic-curve group structure — that are broken in polynomial time by Shor's algorithm on a sufficiently capable fault-tolerant quantum processor. Every transaction record on such a system constitutes a long-lived liability: an adversary may archive signed transactions and public keys today and apply quantum decryption retroactively when hardware of sufficient scale becomes available.

This paper presents BLEEP, a Quantum Trust Network: the first distributed execution protocol in which every block ships with a mathematical proof of its own correctness, every instruction is expressed as intent, and the entire cryptographic foundation survives a quantum computer — by construction, from genesis.

BLEEP enforces transaction validity, node identity, network message authentication, and zero-knowledge proof verification using NIST-finalized post-quantum primitives at Security Level 5. Block validity proofs and cross-chain bridge proofs use Winterfell STARK proofs — transparent, hash-based constructions requiring no trusted setup ceremony. At Protocol Version 5, the prover and verifier are wired to BlockValidityProver and BlockValidityVerifier, benchmarked against the 3,000 ms slot budget, and covered by a 72-hour adversarial test suite.

Four properties distinguish BLEEP from every existing Layer 1:

- Proven execution: every block proposal includes a Winterfell STARK validity proof before broadcast.

- Intent-native runtime: a 6-layer PAT engine and 7-tier VM router resolve caller-declared outcomes to optimal execution paths.

- Quantum-native foundation: SPHINCS+-SHAKE-256f-simple (FIPS 205) and Kyber-1024/ML-KEM-1024 (FIPS 203) are the only primitives on sensitive paths — not a retrofit, the original design.

- Constitutional integrity: maximum supply, inflation ceiling, finality threshold, and fee burn floor are enforced by Rust const_assert! and cannot be altered by any actor.

1. Introduction

1.1 The Harvest-Now, Decrypt-Later Problem

Shor's algorithm, executed on a sufficiently large fault-tolerant quantum processor, reduces integer factorization and discrete-logarithm computation to polynomial time [1]. This breaks RSA, finite-field Diffie-Hellman, and all elliptic-curve schemes, including the secp256k1 curve used by Bitcoin and Ethereum.

The operationally significant threat is archival. Every transaction broadcast on a classical blockchain is a permanent public record. An adversary can archive ciphertexts and signed transactions now and apply quantum decryption retroactively when capable hardware becomes available. This is the harvest-now, decrypt-later threat model [3]. The historical record of a classical blockchain is a cryptographic liability that grows monotonically with time.

1.2 The Migration Problem

A protocol that launches with classical cryptography and plans a post-quantum migration inherits a coordination problem that history shows cannot be cleanly solved. Validators, wallets, bridges, indexers, and all tooling must upgrade simultaneously. HTTPS migration took over a decade and is still not complete [4].

BLEEP eliminates this problem by not having it. Post-quantum from genesis means no migration coordination, no ecosystem split, and no retroactive liability.

1.3 Beyond Security: The Execution Problem

Every existing Layer 1 asks validators and users to trust that a block's state transition was computed correctly. There is no independently verifiable proof. When a block is accepted, it is accepted on the basis of validator signatures — not mathematical proof of execution validity.

BLEEP solves both problems simultaneously. Every block includes a Winterfell STARK validity proof generated before broadcast, verified independently by every validator. Block correctness is not voted upon — it is proven.

1.4 Design Goals

- Post-quantum security at Security Level 5 on all signature, key-encapsulation, and zero-knowledge proof paths, with no classical fallback.

- Proven execution: every block ships with a Winterfell STARK validity proof, generated before broadcast and verified before any vote is cast.

- Intent-native runtime: callers express outcomes, not instructions; the VM router resolves execution path automatically.

- Deterministic protocol execution: byte-identical outputs on every honest node running the same software version.

- Constitutional parameter immutability: enforced by Rust const_assert! — cannot be altered by governance vote or software upgrade.

- Modular separation of concerns: 19 crates with acyclic dependency graphs enforced at build time.

- Trustless cross-chain verification through a tiered bridge architecture requiring no permanently privileged operator.

- Auditability by default: every security-relevant event committed to a tamper-evident, restart-persistent audit log.

2. Background

2.1 Post-Quantum Cryptography

Post-quantum cryptography refers to constructions believed to resist attacks by both classical and quantum computers. Grover's algorithm provides a quadratic speedup against unstructured search — a weakening addressed by larger output sizes. Shor's algorithm provides an exponential speedup against integer factorization and discrete logarithm, rendering RSA and all elliptic-curve schemes insecure regardless of parameter size [5].

In 2024, NIST finalized its first post-quantum cryptography standards [6]. BLEEP uses FIPS 205 (SLH-DSA / SPHINCS+) for signatures and FIPS 203 (ML-KEM / Kyber-1024) for key encapsulation — both at Security Level 5.

2.2 Byzantine Fault Tolerance

A BFT consensus protocol operates correctly when up to f < n/3 participants behave arbitrarily [7]. BLEEP uses proof-of-stake BFT where stake weight replaces uniform vote weight. Finality requires more than two-thirds of total staked supply to commit — not two-thirds of participant count.

2.3 Zero-Knowledge Proofs and Winterfell

A zero-knowledge proof allows a prover to convince a verifier of a statement's truth without revealing information beyond the truth of the statement [8]. BLEEP uses Winterfell STARK proofs for block validity and cross-chain bridge verification. STARKs are constructed over hash functions rather than elliptic curves, providing transparency (no trusted setup) and post-quantum security (security reduces to collision resistance of BLAKE3/SHA3-256).

At Protocol Version 5, proof generation averages ~850 ms on reference hardware (8-core, 32 GB RAM) — within the 3,000 ms slot budget. Determinism has been verified: identical inputs produce byte-identical proofs across all seven testnet validators.

2.4 Intent-Centric Execution

Traditional smart contract platforms require callers to specify exact bytecode execution paths. Intent-centric execution inverts this: callers declare what they want and the protocol resolves how it executes. BLEEP's PAT engine and VM router implement native intent resolution at the protocol layer, not as an application-layer abstraction.

3. System Overview

3.1 Definition: Quantum Trust Network

*A Quantum Trust Network is a distributed execution system in which transaction validity, node identity, network message authentication, and zero-knowledge proof verification are enforced exclusively using cryptographic primitives believed to resist attacks by both classical PPT adversaries and quantum QPT adversaries equipped with Shor**'**s algorithm, as formalized in NIST FIPS 203 and FIPS 205, and in hash-based transparent proof systems.*

3.2 The Four Pillars

Pillar	**What it means**	**How it is implemented**
Proven Execution	Every block ships with a cryptographic proof of correctness	Winterfell STARK BlockValidityProof — generated pre-broadcast, verified pre-vote
Intent Native	Callers express outcomes, not instructions	PAT engine + 7-tier VM router — intent resolved to optimal execution path
Quantum Foundation	No classical public-key primitive on any sensitive path	SPHINCS+ (FIPS 205) + Kyber-1024 (FIPS 203) at Security Level 5 — from block

zero |
| Constitutional Integrity | Supply cap, inflation, finality cannot be changed by anyone | Rust const_assert! — violations do not compile |

Table 1 — The four defining properties of BLEEP

3.3 High-Level Architecture

Subsystem	**Primary Crates**	**Responsibility**
Cryptographic	bleep-crypto, bleep-zkp, bleep-wallet-core	PQ signatures, key encapsulation, STARK proofs, key lifecycle
Consensus	bleep-consensus, bleep-scheduler	Block production, STARK proof pipeline, finality, slashing, epochs
State and Storage	bleep-state, bleep-indexer	Sparse Merkle Trie, RocksDB, shard lifecycle, self-healing
Execution	bleep-vm, bleep-pat, bleep-ai	7-tier VM, intent resolution, deterministic AI advisory
P2P Network	bleep-p2p, bleep-rpc, bleep-auth	Node discovery, gossip, onion routing, authentication

Table 2 — Principal subsystems and primary crates

4. Design Principles

4.1 Safety over Liveliness

Where a choice must be made between safety and liveliness, BLEEP chooses safety. A node that cannot make progress safely halts rather than diverging. This follows the Fischer-Lynch-Paterson result [9]: no deterministic protocol can simultaneously guarantee safety, liveliness, and fault tolerance under asynchrony.

4.2 Determinism as a Protocol Invariant

Every computation influencing chain state must produce byte-identical outputs on every honest node running the same software version. Non-determinism on any of these paths is a protocol bug. STARK proof generation satisfies this invariant: identical public inputs and witnesses produce byte-identical proof serialisations across all validators.

4.3 Constitutional Immutability

Parameter	**Value**	**Enforcement**
Maximum supply	200,000,000 BLEEP	MAX_SUPPLY Rust const_assert! in tokenomics.rs
Minimum finality threshold	>6,667 bps	FinalityManager enforcement
Maximum per-epoch inflation	500 bps (5%)	MAX_INFLATION_RATE_BPS const_assert!
Fee burn floor	2,500 bps (25%)	distribution.rs const_assert!

Table 3 — Constitutional parameters (enforced at compile time)

4.4 Separation of Concerns

Each of the 19 workspace crates has a single defined responsibility. The inter-crate dependency graph is acyclic, enforced at build time. A change to the execution environment cannot inadvertently modify cryptographic behaviour; a vulnerability in networking cannot directly access private key material.

4.5 Auditability by Default

Every security-relevant event is written to a tamper-evident audit log backed by

RocksDB with synchronous writes (sync=true). Log entries are SHA3-256 Merkle-chained. The log survives node restarts: on startup, the chain tip and sequence counter are restored and the most recent 10,000 entries warm the in-memory cache.

5. Architecture

5.1 Cryptographic Subsystem

The cryptographic subsystem (bleep-crypto, bleep-zkp) is the root dependency of the protocol. It provides: SPHINCS+-SHAKE-256f-simple signatures; Kyber-1024/ML-KEM-1024 key encapsulation; AES-256-GCM symmetric encryption; SHA3-256 for state commitments, Merkle hashing, block hashing, and audit log chaining; BLAKE3 for high-throughput content-addressing; and Winterfell STARK proofs for block validity and cross-chain bridge verification. All secret key types are wrapped in zeroize::Zeroizing<Vec<u8>>, zeroed on drop.

5.2 Consensus Subsystem

bleep-consensus implements PoS-BFT in three deterministic modes. PoS-Normal is primary: block production at 3,000 ms intervals with stake-proportional proposer selection. BlockProducer selects up to 4,096 transactions per slot by fee, computes the Sparse Merkle Trie root, generates a Winterfell STARK block validity proof via BlockValidityProver, and signs the completed block with SPHINCS+ before broadcasting.

5.3 State and Storage Subsystem

Account state is maintained as a 256-level Sparse Merkle Trie backed by RocksDB. The trie root appears in every block header. Membership and non-membership proofs are fixed-size at 8,192 bytes regardless of account count. Three RocksDB column families serve security-critical functions: audit_log, audit_meta, and nullifier_store (WriteBatch sync=true).

5.4 Execution Environment

Tier	**Engine**	**Scope**	**Gas Model**
1	Native	BLEEP Transfer, stake, unstake, governance vote	None
2	Router	Intent parsing, engine selection, circuit breakers	Validation only
3	EVM (revm)	Ethereum-compatible contract execution	Ethereum gas semantics
4	WASM (Wasmer + Cranelift)	WASM contract execution	Configurable fuel metering
5	STARK (Winterfell)	ZK execution, public input verification	Fixed cost per verifier op
6	AI-Advised	Constraint validation — advisory, off-chain	Deterministic; no gas
7	Cross-Chain	BLEEP Connect Tier 4 instant intent dispatch	Protocol fee in bps

Table 4 — Execution engine dispatch (source: bleep-vm/src/router/vm_router.rs)

5.5 Peer-to-Peer Network

bleep-p2p uses Kademlia DHT with k=20. Peer IDs are deterministic hashes of SPHINCS+ public keys. All inter-node messages are SPHINCS+-signed; unauthenticated messages are dropped before payload processing. A 2 MiB size gate is enforced at the receive boundary before any deserialization. Onion routing provides multi-hop anonymised delivery using AES-256-GCM keyed from Kyber-1024 per-hop shared secrets.

6. Cryptographic Model

6.1 Algorithm Selection and Rationale

Property	**SPHINCS+-SHAKE-256f-simple**	**Kyber-1024 (ML-KEM-1024)**
NIST Standard	FIPS 205 (SLH-DSA)	FIPS 203 (ML-KEM)
Role	Transaction signing, block signing, P2P auth	Validator binding, peer KEM, onion routing
Security Assumption	One-wayness of SHAKE-256 (hash-based)	Hardness of Module-LWE (lattice-based)
Security Level	Level 5 (≥256-bit post-quantum)	Level 5 (≥256-bit post-quantum)
Public Key	32 bytes	1,568 bytes
Secret Key	64 bytes (Zeroizing on drop)	3,168 bytes (Zeroizing on drop)
Output	7,856-byte detached signature	1,568-byte ciphertext + 32-byte shared secret

Table 5 — Post-quantum algorithm parameters

6.2 STARK Proof System

BLEEP uses Winterfell STARK proofs for block validity and cross-chain bridge verification. STARKs provide transparency (no trusted setup) and post-quantum security (security reduces to collision resistance of BLAKE3 and SHA3-256). At Protocol Version 5:

- winterfell::Prover::prove() is wired to BlockValidityProver, constructing a 48-column execution trace over five public inputs: block_index, epoch_id, tx_count, merkle_root_hash, and validator_pk_hash.

- winterfell::verify() is wired to BlockValidityVerifier and called on every received block proposal before vote broadcast.

- No structured reference string or trusted setup ceremony is required.

Metric	**Value**	**Condition**
Proof generation (avg)	~850 ms	8-core, 32 GB RAM reference hardware
Proof generation (p99)	~1,200 ms	8-core, 32 GB RAM reference hardware
Proof verification (avg)	~12 ms	Same hardware
Slot budget	3,000 ms	Block interval
Margin (avg)	2,150 ms	Budget minus average generation
Trace columns	48	BlockValidityAir
Trusted setup	None	Fully transparent

Table 6 — Winterfell STARK proof timing (Protocol Version 5, Sprint 9)

6.3 The Post-Quantum Boundary

All operations within the boundary are post-quantum secure. No classical public-key primitive (RSA, ECDSA, x25519, BLS) is present on any cryptographically sensitive path. No trusted setup is required for any proof system.

- ✅ Transaction signing — SPHINCS+-SHAKE-256f-simple (FIPS 205, SL5)

- ✅ Block signing — SPHINCS+-SHAKE-256f-simple (FIPS 205, SL5)

- ✅ P2P message authentication — SPHINCS+-SHAKE-256f-simple (FIPS 205, SL5)

- ✅ Key encapsulation — Kyber-1024 / ML-KEM-1024 (FIPS 203, SL5)

- ✅ Block validity proofs — Winterfell STARK (hash-based, no ECC)

- ✅ Cross-chain bridge proofs — SPHINCS+-bound STARK transcripts

- ✅ Identity proofs — SHA3-256 Sparse Merkle Trie paths

7. Execution Model

7.1 State Transition Semantics

STATE TRANSITION FUNCTION: Let S_t denote the complete protocol state at block index t, and let T = (t1, t2, …, tn) be the canonically ordered sequence of validated transactions in block t. The protocol defines a deterministic total function F such that S_{t+1} = F(S_t, T). Given identical S_t and identical T, every correct protocol implementation produces identical S_{t+1}, including the Sparse Merkle Trie root commitment recorded in the block header.

7.2 Intent-Centric Execution

BLEEP does not expose raw bytecode at the API surface. All operations are expressed as typed intent structs. The VM Router (Tier 2) determines the optimal execution engine. Callers declare what they want; the protocol resolves how it executes. The unified gas model normalises costs across all engines, preventing attackers from exploiting cheaper VMs for denial-of-service.

7.3 Transaction Lifecycle

A transaction enters through POST /rpc/tx/submit or P2P mempool gossip. The mempool applies four sequential filters: nonce validity, balance sufficiency, minimum base fee (1,000 microBLEEP), and SPHINCS+ signature verification. BlockProducer selects by fee in descending order up to 4,096 per block. A Winterfell STARK BlockValidityProof is generated before broadcast.

7.4 Block Validity Proofs

BlockValidityCircuit generates STARK proof constraints proving: (a) the block hash is SHA3-256 of its fields; (b) the proposer knows the secret key whose hash equals validator_pk_hash; (c) the epoch ID is consistent with block index and blocks_per_epoch; (d) the SMT root commitment is non-zero. Both the STARK proof and the SPHINCS+ block signature are required for a block to be accepted.

7.5 StateDiff Isolation

The VM never writes to bleep-state directly. Execution produces a StateDiff object committed atomically by bleep-state only after validator quorum. This guarantees clean separation between execution and state commitment, dry-run simulation without side effects, and deterministic rollback safety.

8. Networking and Consensus

8.1 Validator Model and Fault Assumptions

*VALIDATOR SET AND FAULT MODEL: Let V = {v1, …, vn} be the active validator set at epoch e. Each vi carries a SPHINCS+ verification key vki, a Kyber-1024 encapsulation key eki, and a stake si in microBLEEP. Total staked supply S = Σ(si). Safety is guaranteed when Byzantine stake f **<** S/3.*

8.2 Three Deterministic Consensus Modes

Mode	**Trigger**	**Behaviour**
PoS-Normal	Primary — healthy validator set	Stake-proportional proposer selection, 3,000 ms slots
Emergency (PBFT)	<67% validators responsive	Reduced validator set, safety-first
Recovery (PoW)	Post-partition re-anchor	Deterministic re-sync from last finalised block

Table 7 — Consensus modes

8.3 Consensus Protocol Flow

- Proposer selection: at each 3,000 ms slot boundary, a validator is selected with probability proportional to stake fraction.

- Block proposal: the proposer generates a Winterfell STARK proof, signs with SPHINCS+, and broadcasts.

- Block validation: each validator independently verifies the STARK proof, SPHINCS+ signature, and SMT root transition.

- Vote: accepting validators broadcast a SPHINCS+-signed prevote, then a signed precommit.

- Finalisation: a block is finalised when precommits representing >6,667 bps of total staked supply are received. Irreversible.

- Epoch transition: every 1,000 blocks (mainnet) / 100 blocks (testnet), the validator set rotates and rewards are distributed.

8.4 Slashing

Violation	**Penalty**	**Source**
Double-sign	33% of stake burned; validator tombstoned	double_signing_penalty: 0.33
Equivocation	25% of stake burned	equivocation_penalty: 0.25
Downtime	0.1% per consecutive missed block	downtime_penalty_per_block
Tier 4 bridge executor timeout	30% of executor bond	EXECUTION_TIMEOUT = 120 s

Table 8 — Slashing parameters (source: bleep-consensus/src/slashing_engine.rs)

9. Governance

9.1 Proposal Lifecycle

LiveGovernanceEngine processes typed proposals through a six-stage lifecycle: Submit → AIConstraintValidator pre-flight → Active → Tally → Execute → Record. A proposal that would set MaxInflationBps above 500 is rejected at pre-flight and never reaches a vote.

Parameter	**Testnet Value**	**Notes**
voting_period_blocks	1,000 blocks (~50 min)	At 3-second block time
quorum_bps	1,000 bps (10%)	Minimum stake participation
pass_threshold_bps	6,667 bps (66.67%)	Yes votes of participating stake
veto_threshold_bps	3,333 bps (33.33%)	Veto votes to block passage
min_deposit	10,000 BLEEP	Minimum to submit a proposal

Table 9 — LiveGovernanceEngine configuration

9.2 Zero-Knowledge Voting

ZKVotingEngine provides privacy-preserving stake-weighted voting. Votes are cast as EncryptedBallot structs. EligibilityProof establishes voting power without revealing validator identity. TallyProof enables independent tally verification without revealing individual votes. Three voter roles: Validator (1.0×), Delegator (0.5×), Community token holder (0.1×).

9.3 Constitutional Constraints

Four parameters are constitutionally immutable, enforced by Rust const_assert!. A code change that violates them does not compile. They cannot be altered by any governance vote, software upgrade, or validator supermajority.

9.4 Forkless Protocol Upgrades

ForklessUpgradeEngine manages hash-committed, deterministic protocol upgrades activating at epoch boundaries only. Version progression is monotonically enforced — a version mismatch halts the chain. No node restart required. Partial upgrades are rejected atomically.

10. Cross-Chain Interoperability

BLEEP Connect is a four-tier cross-chain bridge architecture implemented across ten sub-crates within bleep-interop. No tier requires a permanently privileged operator or a trusted multisig key set.

10.1 Bridge Tier Overview

Tier	**Protocol**	**Latency**	**Security Basis**	**Status**
4 — Instant	Executor auction + escrow	200ms – 1s	Economic: 30% bond slashed on timeout	✅ Ethereum Sepolia
3 — ZK Proof	SPHINCS+-bound STARK commitment	10–30s	Cryptographic: PQ-secure, zero trusted operators	✅ Ethereum Sepolia
2 — Full-Node	Multi-client verification (≥3 nodes)	Hours	90% consensus across independent nodes	🔲 Mainnet target
1 — Social	Stakeholder governance	7 days / 24h emergency	Full governance consensus	🔲 Mainnet target

Table 10 — BLEEP Connect bridge tiers

10.2 Tier 3: ZK Proof Bridge

The Tier 3 bridge batches up to 32 cross-chain intents into proof bundles. ProofGenerator constructs a deterministic transcript, binds it with a SPHINCS+ signature, and generates a Winterfell STARK commitment. ProofVerifier verifies using the post-quantum public key and the Winterfell verifier. No structured reference string or MPC ceremony is required. Double-spend prevention uses GlobalNullifierSet with atomic WriteBatch (sync=true).

11. Economics and Tokenomics

11.1 Token Parameters

Parameter	**Value**	**Source**
Maximum supply (†)	200,000,000 BLEEP	MAX_SUPPLY in tokenomics.rs
Token decimals	8 (1 BLEEP = 10^8 microBLEEP)	tokenomics.rs
Initial circulating supply	25,000,000 BLEEP (12.5%)	INITIAL_CIRCULATING_SUPPLY
Maximum per-epoch inflation (†)	500 bps (5%)	MAX_INFLATION_RATE_BPS
Fee burn split (†)	2,500 bps (25%)	FEE_BURN_BPS in distribution.rs
Validator fee split	5,000 bps (50%)	FEE_VALIDATOR_REWARD_BPS
Treasury fee split	2,500 bps (25%)	FEE_TREASURY_BPS
Minimum base fee	1,000 microBLEEP	MIN_BASE_FEE in fee_market.rs

Table 11 — Token parameters († = constitutional)

11.2 Token Distribution

Allocation	**Tokens**	**%**	**Launch Unlock**	**Vesting**

Validator Rewards	60,000,000	30%	10,000,000	Emission decay schedule
Ecosystem Fund	50,000,000	25%	5,000,000	10-year linear; governance disbursement
Community Incentives	30,000,000	15%	5,000,000	Governance-triggered release
Foundation Treasury	30,000,000	15%	5,000,000	6-year linear; governance spending
Core Contributors	20,000,000	10%	0	1-year cliff + 4-year linear; immutable on-chain
Strategic Reserve	10,000,000	5%	0	Governance-controlled unlock

Table 12 — Token distribution

11.3 Validator Emission Schedule

Year	**Rate**	**Annual Emission (BLEEP)**	**Cumulative**	**Pool Remaining**
1	12%	7,200,000	7,200,000	52,800,000
2	10%	6,000,000	13,200,000	46,800,000
3	8%	4,800,000	18,000,000	42,000,000
4	6%	3,600,000	21,600,000	38,400,000
5+	4%	~2,400,000/yr	—	Decreases annually

Table 13 — Validator emission schedule

11.4 Game-Theoretic Safety Verifier

SafetyVerifier in bleep-economics/src/game_theory.rs evaluates five attack models: Equivocation, Censorship, NonParticipation, Griefing, and Cartel formation. A CI build fails if any model returns is_profitable = true at current parameters — machine-verified economic safety analogous to the compile-time constitutional invariants.

12. AI Advisory and Inference Engine

bleep-ai provides two systems. Neither participates in block production, consensus voting, or any state-modifying operation without a prior governance vote. AI outputs are inputs to the governance process, not outputs of it.

12.1 Phase 3: Rule-Based Advisory

AIConstraintValidator is a deterministic rule engine that checks governance proposals against the four constitutional invariants before they enter the vote queue. Not a trained model — it applies explicit rules. A proposal setting MaxInflationBps above 500 is rejected with a descriptive error before any vote is cast.

12.2 Phase 4: DeterministicInferenceEngine

DeterministicInferenceEngine is an ONNX-based inference runtime enforcing six invariants: SHA3-256 model hash verification, deterministic input normalisation, deterministic output rounding, CPU-only execution, governance-approval gating, and no dynamic model loading. Every inference produces an auditable InferenceRecord queryable at GET /rpc/ai/attestations/{epoch}.

12.3 AI Safety Scope Boundaries

- AI outputs are advisory only — no write access to chain state or block production pipeline.

- All AI outputs are signed and verifiable via AIAttestationManager.

- AI cannot override governance authority.

- Failed inference returns an explicit, typed error — no silent degradation.

13. Scalability under Deterministic Constraints

13.1 Sharding Model

BLEEP partitions state across 10 shards (NUM_SHARDS) in the testnet configuration. Each shard maintains an independent RocksDB instance and processes transactions in parallel. The shard count is a governance parameter bounded by the BFT safety requirement: each shard must maintain f < S_shard/3.

13.2 Projected Performance

Metric	**Projected Value**
Configuration	10 shards, 4,096 tx/block, 3,000 ms interval, simulated workload
Average TPS	10,921 (target ≥10,000)
Peak TPS	13,200
Sustained minimum TPS	9,840
Full-capacity block ratio	82.3%
STARK proof generation (avg)	~850 ms per block
STARK proof verification (avg)	~12 ms per block

Table 14 — Projected benchmark targets (simulated workload, pre-testnet; STARK timings on reference 8-core, 32 GB RAM hardware)

Important: these figures are projections based on simulated workloads. Actual throughput on a geographically distributed public testnet will be measured and published during Phase 6.

14. Use Cases

14.1 Sovereign Digital Asset Custody

Institutions managing digital assets over multi-decade horizons face retroactive vulnerability from the harvest-now, decrypt-later threat. BLEEP's SPHINCS+ signing ensures that custody records signed at genesis remain computationally opaque against future quantum adversaries. BLEEP's NIST-standardised post-quantum primitives align with EU Critical Infrastructure PQC mandate (2030) and government procurement requirements.

14.2 Cross-Chain Settlement Infrastructure

BLEEP Connect's four-tier architecture allows settlement systems to select the appropriate security level for each transfer. Tier 3 provides cryptographic verification via SPHINCS+-bound STARK commitments with no trusted operator. The nullifier store with atomic writes prevents cross-chain double-spend at the protocol level.

14.3 Verifiable Computation and Proof Markets

Application developers requiring a ZK proof system that remains sound against quantum adversaries can build on BLEEP's Winterfell STARK execution tier — no trusted setup, post-quantum secure. AIAttestationManager creates an on-chain record of each inference including model hash, normalised inputs, and outputs.

14.4 Regulated Financial Infrastructure

BLEEP's constitutional compile-time invariants provide machine-verifiable evidence that supply cap, inflation rate, and fee burn parameters cannot be altered without a code change that fails to compile. The tamper-evident audit log provides non-repudiation records for MiCA, DORA, and SEC digital asset

compliance.

15. Target Users

Institutional Asset Managers and Custodians operating under multi-decade investment horizons require cryptographic guarantees that remain valid regardless of future quantum advances. BLEEP provides SPHINCS+-signed transaction records from genesis with no migration risk.

Regulated Financial Institutions require deterministic settlement finality, non-repudiable audit trails, and governance mechanisms preventing unilateral parameter changes. BLEEP's constitutional compile-time assertions and Merkle-chained audit log satisfy these requirements.

Cross-Chain Protocol Developers require trustless verification without permanently privileged operators. BLEEP Connect's Tier 3 STARK bridge and Tier 4 economic slash-bond design eliminate persistent trust assumptions.

AI-Native Application Developers building autonomous agents need an execution environment where agents express intent rather than manage bytecode. BLEEP's PAT engine is infrastructure for autonomous on-chain coordination.

16. Security Considerations

16.1 Threat Model

BLEEP's security analysis considers three adversary classes: Classical PPT adversary (targets 256-bit security on all operations); Quantum QPT adversary (BLEEP's post-quantum boundary maintains 256-bit security — no path is broken by Shor's algorithm); and Byzantine validator adversary (controls f < S/3 of staked supply — BFT safety holds unconditionally).

16.2 Independent Security Audit

Severity	**Count**	**Resolved**	**Acknowledged**	**Outcome**
Critical	2	2	0	All resolved
High	3	3	0	All resolved
Medium	4	3	1	SA-M4: EIP-1559 design property; documented in THREAT_MODEL.md
Low	3	3	0	All resolved
Informational	2	1	1	SA-I2: NTP drift guard — mainnet gate
Total	14	12	2	Cleared for testnet preparation

Table 15 — Sprint 9 internal audit finding summary

16.3 Adversarial Test Suite

Scenario	**Result**	**Invariant Verified**
ValidatorCrash(1)	✅ Pass	f=1 < 2.33; consensus resumed
ValidatorCrash(2)	✅ Pass	f=2 < 2.33; consensus resumed
NetworkPartition(4/3)	✅ Pass	Majority partition continued; healed cleanly
LongRangeReorg(10)	✅ Pass	Rejected at FinalityManager (I-CON3)
LongRangeReorg(50)	✅ Pass	Rejected at FinalityManager (I-CON3)
DoubleSign(validator-0)	✅ Pass	33% slashed; evidence committed; tombstoned
TxReplay	✅ Pass	Rejected by nonce check (I-S5)
InvalidBlockFlood(1000)	✅ Pass	Rejected at SPHINCS+ gate; peer rate-limited
STARKProofTamper	✅ Pass	Tampered proof rejected at BlockValidityVerifier
LoadStress(10,000 TPS, 60s)	✅ Pass	Max throughput; STARK proofs within

slot budget |

Table 16 — Adversarial test scenarios (72-hour continuous run, Protocol Version 5)

17. Limitations

17.1 Post-Quantum Primitives Introduce Measurable Overhead

SPHINCS+-SHAKE-256f-simple produces 7,856-byte signatures. On a 4,096-transaction block, aggregate signature data is approximately 32 MB, imposing a minimum bandwidth requirement of approximately 87 MB/s from signatures alone. Kyber-1024 public keys are 1,568 bytes compared to 32-byte Curve25519 keys. Winterfell proof generation averages ~850 ms per block. These overheads are the direct, quantified cost of transparent post-quantum security with no trusted setup — accepted as an explicit design trade-off.

17.2 Signature Aggregation Not Yet Available

SPHINCS+ does not support aggregation: n validators produce n independent 7,856-byte signatures. Hash-based signature aggregation is a medium-term research direction (Section 18).

17.3 Throughput Figures Are Simulated

TPS projections reflect simulated conditions: 7 validators, controlled network latency, geographically concentrated nodes, and a uniform transaction mix. Actual throughput will be measured during Phase 6 public testnet operation.

17.4 Ed25519 Present in Cargo.toml

Ed25519 is present for compatibility and is not used on any sensitive path when the quantum feature flag is active (default). It is scheduled for sunset in Phase 9.

18. Future Work

18.1 Public Testnet Expansion

Phase 6 targets ≥50 validators across ≥6 continents, a 30-day sustained run, a public block explorer, and a 100,000 BLEEP bug bounty pool. This milestone validates BFT safety at realistic scale and produces measured (not simulated) performance data.

18.2 Mainnet Deployment

Mainnet requires: ≥21 validators, governance active from block 1, BLEEP Connect Tiers 1–4 on Ethereum and Solana, client SDKs, NTP drift guard active, and GenesisAllocation vesting contracts deployed.

18.3 Signature Aggregation

Hash-based signature aggregation combining Merkle-based multi-signatures with SPHINCS+ would reduce per-block vote bandwidth by O(log n) in validator count — a medium-term research direction.

18.4 Phase 8 Ecosystem Expansion

Move language execution engine, zkEVM compatibility, Cosmos/Polkadot bridge activation, and additional chain adapters via governance vote.

18.5 Phase 9 — Quantum-Safe Mainnet (2028+)

Mandatory SPHINCS+ enforcement across all transaction types, Ed25519 sunset,

Kyber-1024 mandatory for all session key establishment, and bleep-vm BSL-1.1 → Apache 2.0 automatic conversion (2028-07-13).

19. Conclusion

BLEEP is a Quantum Trust Network: the first distributed execution protocol in which every block ships with a mathematical proof of its own correctness, every instruction is expressed as intent, and the entire cryptographic foundation is built on NIST-standardised post-quantum primitives — by construction, from genesis.

No classical public-key primitive or pairing-based construction is present on any cryptographically sensitive path. No trusted setup ceremony is required for any proof system. No migration path is needed — because the problem was solved before the protocol accumulated economic value and ecosystem dependencies.

Protocol Version 5 establishes the practical foundation for this design: SPHINCS+-signed blocks at a 3,000 ms slot interval, Kyber-1024 key encapsulation for all peer channels, Winterfell STARK block validity proofs benchmarked at ~850 ms generation and ~12 ms verification, a 72-hour adversarial test suite with all scenarios passing, an internal security audit with all Critical and High findings resolved, and projected throughput averaging 10,921 TPS across 10 shards under simulated conditions.

The four pillars — proven execution, intent-native runtime, quantum-native foundation, and constitutional integrity — are verifiable properties of the Protocol Version 5 codebase. Public testnet deployment in Phase 6 will validate these properties at the scale and adversarial conditions required for a globally adopted protocol.

References

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

[2] Banegas, G. et al. (2021). Concrete quantum cryptanalysis of binary elliptic curves. IACR Transactions on Cryptographic Hardware and Embedded Systems.

[3] Mosca, M. (2018). Cybersecurity in an era with quantum computers: will we be ready? IEEE Security & Privacy, 16(5), 38–41.

[4] Amann, J. et al. (2017). Mission accomplished? HTTPS security after DigiNotar. ACM IMC 2017.

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

[6] NIST. (2024). Post-Quantum Cryptography Standardization. FIPS 203, FIPS 204, FIPS 205.

[7] Lamport, L., Shostak, R., and Pease, M. (1982). The Byzantine generals problem. ACM TOPLAS, 4(3), 382–401.

[8] Ben-Sasson, E. et al. (2018). Scalable, transparent, and post-quantum secure computational integrity. IACR ePrint 2018/046.

[9] Fischer, M.J., Lynch, N.A., and Paterson, M.S. (1985). Impossibility of distributed consensus with one faulty process. Journal of the ACM, 32(2), 374–382.

[10] Winterfell STARK library. (2024). https://github.com/facebook/winterfell

[11] Bernstein, D.J. and Lange, T. (2017). Post-quantum cryptography. Nature,

549, 188–194.

[12] Buchman, E., Kwon, J., and Milosevic, Z. (2018). The latest gossip on BFT consensus. arXiv:1807.04938.

Appendix A: Protocol Parameters

All values are drawn from the production Rust source at Protocol Version 5. Parameters marked (†) are constitutional and cannot be changed by governance vote or software upgrade.

A.1 Consensus and Execution

Parameter	**Value**	**Source Constant**
Block interval	3,000 ms	BLOCK_INTERVAL_MS
Max transactions per block	4,096	MAX_TXS_PER_BLOCK
Blocks per epoch (mainnet)	1,000	BLOCKS_PER_EPOCH
Blocks per epoch (testnet)	100	testnet-genesis.toml
Finality threshold (†)	>6,667 bps of total stake	FinalityManager
Active shards	10	NUM_SHARDS
Double-sign slash	33% of stake	double_signing_penalty
Equivocation slash	25% of stake	equivocation_penalty
Downtime slash	0.1% per missed block	downtime_penalty_per_block

A.2 Cryptography and Networking

Parameter	**Value**	**Source Constant**
SPHINCS+ signature size	7,856 bytes	SPHINCS_SIG_LEN
SPHINCS+ public key size	32 bytes	pqcrypto-sphincsplus
Kyber-1024 public key size	1,568 bytes	pqcrypto-kyber
State trie depth	256 levels	TRIE_DEPTH
Merkle proof size	8,192 bytes (fixed)	SparseMerkleTrie
Gossip max message size	2,097,152 bytes (2 MiB)	MAX_GOSSIP_MSG_BYTES
Gossip fanout	8	bleep-p2p
Kademlia k-bucket size	20	bleep-p2p
STARK trace columns	48	BlockValidityAir
STARK proof gen time (avg)	~850 ms (reference hardware)	bleep-zkp benchmarks
STARK verify time (avg)	~12 ms (reference hardware)	bleep-zkp benchmarks
Proof setup requirement	None (transparent)	bleep-zkp
JWT entropy minimum	3.5 bits/byte (Shannon)	session.rs

A.3 Economics and Token

Parameter	**Value**	**Source Constant**
Maximum supply (†)	200,000,000 BLEEP	MAX_SUPPLY
Token decimals	8	tokenomics.rs
Initial circulating supply	25,000,000 (12.5%)	INITIAL_CIRCULATING_SUPPLY
Maximum per-epoch inflation (†)	500 bps (5%)	MAX_INFLATION_RATE_BPS
Fee burn split (†)	2,500 bps (25%)	FEE_BURN_BPS
Validator fee split	5,000 bps (50%)	FEE_VALIDATOR_REWARD_BPS
Treasury fee split	2,500 bps (25%)	FEE_TREASURY_BPS
Min base fee	1,000 microBLEEP	MIN_BASE_FEE
Max base fee	10,000,000,000 microBLEEP	MAX_BASE_FEE

A.4 Cross-Chain Bridge

Parameter	**Value**	**Source Constant**
Tier 3 proof type	SPHINCS+-bound STARK commitment	bleep-connect-layer3-zkproof

Tier 3 batch size	32 intents	L3_BATCH_SIZE
Tier 3 setup requirement	None (transparent)	bleep-zkp
Tier 4 execution timeout	120 s	EXECUTION_TIMEOUT
Tier 4 protocol fee	10 bps (0.1%)	PROTOCOL_FEE_BPS
Tier 4 bond slash	30%	EXECUTOR_SLASH_BPS
Tier 2 consensus threshold	90%	CONSENSUS_THRESHOLD
Tier 2 minimum verifiers	3	MIN_VERIFIER_NODES

BLEEP · Quantum Trust Network · Protocol Version 5 · May 2026

github.com/BleepEcosystem/BLEEP-v1 · bleepecosystem.com

© 2026 Muhammad Attahir — bleepecosystem.com