

Boing Blockchain — Implementation Roadmap

Priorities: Security → Scalability → Decentralization → Authenticity

Goal: Build an authentic, efficient L1 with unique innovations.

Aligned with: [BOING-BLOCKCHAIN-DESIGN-PLAN.md](#) (UX, Technical Innovations, Sustainability)

Recommendations: [DEVELOPMENT-AND-ENHANCEMENTS.md](#) (SDK, Automation, dApp Incentives)

Quick Start

```
# Build the workspace
cargo build

# Run the node
cargo run -p boing-node

# Run tests
cargo test
```

Project Structure

```
boing-network/
├─ Cargo.toml           # Workspace root
├─ crates/
│  ├─ boing-primitives/ # Types, hash, crypto, signatures
│  ├─ boing-consensus/  # PoS + HotStuff BFT
│  ├─ boing-state/      # Verkle tree, state store
│  ├─ boing-execution/ # VM, parallel scheduler
│  ├─ boing-tokenomics/ # Block emission, fee split, dApp incentives
│  ├─ boing-governance/ # Phased governance (time locks)
│  ├─ boing-automation/ # Scheduler, triggers, executor incentives
│  ├─ boing-qa/         # Protocol QA (Allow/Reject/Unsure)
│  ├─ boing-cli/        # boing init, dev, deploy
│  ├─ boing-p2p/        # libp2p networking
│  └─ boing-node/       # Node binary
├─ docs/                # All project documentation
├─ fuzz/                # Cargo-fuzz harness (boing-primitives)
└─ website/             # boing.network site (Astro + Cloudflare)
```

Phase 1: Foundation (Weeks 1–8)

1.1 Primitives — COMPLETE ✓

- ✓ Hash , Hasher (BLAKE3)
- ✓ AccountId , Account , AccountState
- ✓ Transaction , TransactionPayload , AccessList
- ✓ Block , BlockHeader

- Ed25519 signature; `SignedTransaction` ; sign/verify
- Unit tests (hash, tx id, signatures, access-list conflicts)

1.2 Consensus — COMPLETE ✓

- HotStuff state machine (Propose → Vote → Commit)
- Quorum ($2f+1$); single-validator and multi-validator modes
- Unit tests (propose_and_commit, propose + vote flow)
- Leader rotation (round-robin)
- Simulate 4 nodes, 1 Byzantine

1.3 State Store

- In-memory `StateStore` (HashMap)
- Sparse Merkle tree for state root
- Transaction root (Merkle tree of tx IDs via `tx_root`)
- State root from Sparse Merkle
- Apply/revert batch (checkpoint/revert)

1.4 Execution (Minimal)

- VM: `Transfer` execution
- `TransactionScheduler` (access-list batching)
- Unit tests (scheduler batches)
- `BlockExecutor` (execute block of txs)
- Gas metering stub (fixed per tx type)
- Nonce validation

Phase 2: Networking & Integration (Weeks 9–16)

2.1 P2P Layer

- libp2p swarm setup (TCP, noise, yamux, TLS)
- Gossip: blocks, transactions (gossipsub)
- Peer discovery (mDNS)
- Request/response: get blocks by hash/height
- Block propagation and import (full node receives via gossip)

2.2 Advanced Decentralized Peer Discovery (see [DECENTRALIZATION-AND-NETWORKING.md](#))

- DHT-based discovery (Kademlia); minimize fixed bootnode reliance
- Gossip-first overlay: random peer selection, active probing
- Bootnode rotation (governance or community-funded)
- Peer scoring/reputation for Sybil/eclipse resistance
- WebRTC/WebSockets for browser light clients; decentralized signaling
- Incentivized relayers; DHT rendezvous for NAT traversal

2.2b Decentralized WebRTC Signaling (see [DECENTRALIZATION-AND-NETWORKING.md](#))

- Signaling smart contract (offer/answer exchange, event emission)
- Spam prevention: on-chain rate limiting, deposit/stake for offers, identity/reputation gate
- IPFS/Filecoin integration for large SDP payloads; on-chain CID pointers
- DHT announcement of WebRTC capability; recipient lookup
- Incentivized STUN/TURN servers; protocol rewards for relay providers
- STUN/TURN on-chain registry with staking; reputation score from uptime, latency, success rate
- STUN/TURN slashing for poor performance; SDK server selection by reputation

2.3 Node Integration

- Mempool (transaction pool)
- Wire consensus ↔ execution ↔ state
- Block production loop (BlockProducer + produce_block_if_ready)
- Chain state (ChainState, genesis, append)
- CLI: `--validator` , `--rpc-port` , `--data-dir`
- JSON-RPC HTTP server (boing_submitTransaction, boing_chainHeight)
- Validator loop (produce blocks every BLOCK_TIME when --validator)
- Block import and validation (validate_and_execute_block, import_block)

2.4 End-to-End

- Single node: produce blocks with txs
- Two nodes: sync blocks (P2P gossip + block import)
- Multi-node testnet (4+ validators)

Phase 3: Verkle & VM (Weeks 17–28)

3.1 Verkle Tree / Sparse Merkle

- Sparse Merkle tree (insert, get, delete, root)
- State root from Sparse Merkle
- Proof generation (MerkleProof, prove/verify)
- Stateless client verification (boing_getAccountProof, boing_verifyAccountProof RPC)

3.2 Custom VM

- Bytecode spec (PUSH, ADD, SUB, MUL, MLOAD, MSTORE, SLOAD, SSTORE, JUMP, JUMPI, RETURN, STOP)
- Interpreter (deterministic stack machine)
- Gas metering per opcode
- Contract deployment and call

3.3 Parallel Execution (Full)

- Execute Transfer-only batches in parallel (rayon)
- Conflict detection at runtime (sanity check)
- Benchmarks: sequential vs parallel (criterion)

Phase 4: PoS & Production Readiness (Weeks 29–40)

4.1 Proof of Stake

- Validator set from stake (`StateStore::top_stakers`)
- Staking transactions (bond, unbond)
- Slashing conditions (equivocation detection in consensus)
- Rewards distribution (block emission to proposer)
- Sustainable tokenomics (design + boing-tokenomics crate: emission curve, fee split, burn, dApp cap)

4.2 Core Innovations (from Design Plan)

- Native Account Abstraction (protocol-level)
- Adaptive gas model (GasConfig, multiplier with predictable caps)
- Phased governance (boing-governance: proposal → cooling → execution)
- Transparent slashing with appeal

4.3 Hardening

- Fuzz testing (cargo-fuzz; fuzz/ crate, bincode Block/Transaction)
- Property-based tests (proptest: balance preservation, parallel vs sequential equivalence)
- Security audit (external); continuous audit cadence post-mainnet
- Documentation, runbook

4.4 Randomness & Fair Ordering

- VDF/VRF for verifiable randomness (leader selection; `leader_from_vrf`, `dummy_vrf_output`)
- Replace or augment round-robin with VDF-driven selection (`leader_from_vrf` stub)

4.5 Security Standards (see [SECURITY-STANDARDS.md](#))

- DDoS resistance: RPC rate-limiting (`RateLimitConfig`, governor crate)
- Disk persistence (chain + state via `--data-dir`)
- P2P connection cap per remote IP (`RateLimitConfig.connections_per_ip` , `--max-connections-per-ip` ; signed tx gossip + mempool admit on receive — see [RUNBOOK.md](#) §8.1)
- Formal verification pipeline for critical contracts (staking, DeFi, automation)
- Bug bounty program
- Incident response runbook ([RUNBOOK.md](#) §6)
- Post-quantum cryptography research and integration path
- HD wallet path type (`HdPath` in boing-primitives)

4.6 Protocol Quality Assurance (QA) — see [QUALITY-ASSURANCE-NETWORK.md](#)

- **boing-qa crate:** Rule types (Allow / Reject / Unsure), central rule registry, deterministic checks (bytecode size, opcode whitelist, well-formedness, purpose categories including meme/community/entertainment)
- **Known edge-case resolution:** Implement §11 mappings so automation handles meme leniency, ambiguous declaration, new/unknown pattern; blocklist support
- **Node integration:** Run QA at submit and/or mempool insert for `ContractDeploy`; structured rejection (`rule_id`, message); optional `boing_qaCheck` RPC
- **Execution defense in depth:** QA check in `execute_contract_deploy` before `set_contract_code`

- **Community QA pool:** Pending QA queue (`boing_qa::pool`), pool members, voting, max time T, default outcome at expiry; on-chain integration pending
 - **Governance of rules:** Add/modify rules via time-locked governance; versioned rule sets
 - **Purpose in payload:** `ContractDeployWithPurpose` variant; `check_contract_deploy_full` with `RuleRegistry`
 - **Always-review categories:** Policy categories that force Unsure; configurable via `RuleRegistry`
 - **Scam patterns:** Byte-sequence blacklist (legitimacy heuristic) in `RuleRegistry`
 - **Soft rules:** Example: "other" + minimal description → Unsure
 - **Additional enhancements (optional):** Pre-flight SDK/CLI (`boing_qaCheck` RPC + SDK `client.qaCheck()`), public QA metrics, appeal path, canonical malice definition, deployer checklist (in progress)
-

Phase 5: UX & Human-Centered Innovations (Weeks 41–52)

5.1 Frictionless UX

- Transaction simulation API (`boing_simulateTransaction`)
- Gasless transactions (paymaster / meta-tx types: `PaymasterConfig`, `SponsoredTransaction`)
- Human-readable signing (`display_for_signing` on `SignedTransaction`, `SignedIntent`)
- Transaction simulation API (pre-flight)
- Pre-confirmation / outcome guarantees where feasible
- Automatic chain routing (cross-chain UX)

5.1b Intent-Based Execution (see [DEVELOPMENT-AND-ENHANCEMENTS.md](#))

- Intent signing format (`SignedIntent`, `IntentPool`, `boing_submitIntent` RPC)
- Meta-router: orchestrate optimal cross-chain path
- Solver/executor integration for intent fulfillment

5.2 Trust & Verification

- Intent-based transaction format (`SignedIntent`, `boing_submitIntent`)
- Simulation service / SDK
- Contract/address attestation hooks
- Verified display for addresses and amounts

5.3 Recoverability

- Social recovery types (`Guardian`, `RecoveryRequest` in `boing-primitives`)
- Protocol-level recovery flows
- Optional MPC / threshold custody integration points

5.4 MEV & Cost Predictability

- VDF-based fair ordering (or equivalent)
- Encrypted mempool options
- Predictable gas model
- MEV-aware transaction design

5.5 Developer Experience

- Boing CLI (`boing init` , `boing dev` , `boing deploy`)
- RPC API specification ([RPC-API-SPEC.md](#))
- Local/devnet parity for testing
- Cross-chain simulation support

5.6 Success-Based dApp Incentives & Boing SDK

- dApp incentive formula (DappMetrics, `dapp_incentive`, `VALUE_CAP_PER_DAPP`)
- Per-dApp owner cap (governance parameter)
- Success metrics (tx count, fees, volume)
- Automated payout distribution
- Dynamic fee allocation & developer royalties (dApp-specified splits)
- Reputation-based resource access (priority processing, discounted gas)
- Boing SDK (`boing init` , `boing dev` , `boing deploy`)
- Metrics registration API

5.6b Boing Studio & AI-Assisted SDK

- Boing Studio IDE (Remix-style: templates, debugging, one-click deploy)
- AI-assisted SDK (code generation, vulnerability scanning, optimization)

5.7 Decentralized Automation

- Native scheduler (CronSchedule, Scheduler)
- Execution verification (ExecutorAttestation, ZkpProof, FraudProof, OracleAttestation)
- Trigger-based execution hooks
- Decentralized executor incentives (`executor_reward`, `executor_slash` in `boing-automation`)
- User-facing automation SDK

Phase 6: Technical Innovations for Ecosystem (Weeks 53–64+)

6.1 Cross-Chain Primitives

- Light client verification at protocol level
- Bridge standards (IBC-style or custom)
- Cross-chain messaging primitives
- Trust-minimized bridge design (ZKP/MPC relayers with slashing)
- Decentralized oracle networks for external data

6.2 Stateless & Light Clients

- Verkle proof generation and verification (MerkleProof)
- Light client sync protocol
- Mobile-friendly validation
- Compact proof specs (for ecosystem adoption)
- Browser-based Wasm light clients

6.3 Intent & Advanced Execution

- Intent signing format (SignedIntent, IntentPool, `boing_submitIntent`)

- Solver/executor integration points
- Pre-confirmation API
- Commit-reveal for MEV-sensitive ops

6.4 Compliance & Real-World

- Privacy-preserving attestation design
- Off-ramp / fiat integration points
- Payment rail primitives
- Compliance-friendly APIs (selective disclosure)

6.5 Open Standards

- RPC/API spec ([RPC-API-SPEC.md](#))
- Bridge protocol documentation
- Reference implementations for interoperability

6.6 Decentralized Storage & Network Monitoring

- Filecoin/Arweave integration for permanent archival (chain state, tx logs)
- SDK tools for IPFS/Filecoin dApp frontend deployment
- Network topology monitoring (node distribution, decentralization dashboards)

Phase 7: Testnet & Mainnet (Ongoing from Phase 4)

7.1 Public Testnet

- Public testnet launch
- Testnet incentive program (sustainability-first): use [TESTNET.md](#) Part 3 for readiness checklist, incentive design (validators, developers, users), and 2–4 week duration.
- Validator onboarding
- boing.finance integration (testnet)
- Community engagement: grant programs, hackathons, educational content

7.2 Mainnet

- Token distribution (fair launch)
- Bridge to boing.finance and other chains
- Mainnet launch
- Governance activation

Innovation Summary (from Design Plan)

| Category | Key Items |
|--------------|---|
| UX | Native AA, gasless, human-readable signing, simulation, social recovery |
| Trust | Intent display, attestation, verified addresses |
| MEV | VDF ordering, encrypted mempool, predictable gas |

| | |
|-----------------------|---|
| Technical | Verkle stateless clients, light-client bridges, intent architecture |
| Ecosystem | Open standards, bridge protocols, reusable specs |
| Sustainability | Emission curve, fee-dominant revenue, no reward cliffs |

Boing Network — Authentic. Decentralized. Optimal. Sustainable.