AGNTC Whitepaper

ZK Agentic Chain

A Privacy-Preserving Blockchain with
AI-Powered Verification

Version 1.2 | March 2026

zkagentic.ai

Table of Contents

Abstract

We present ZK Agentic Chain, a Layer-1 blockchain protocol that introduces Proof of AI Verification (PoAIV) - a consensus mechanism in which autonomous AI agents verify chain integrity through zero-knowledge private channels. Unlike traditional proof-of-work systems that consume energy solving arbitrary hash puzzles, or proof-of-stake systems that concentrate power among the wealthiest token holders, PoAIV selects a committee of 13 AI verification agents per block, requiring a 9/13 supermajority attestation threshold for consensus. Verification agents apply reasoning to their audits - examining logical consistency, cross-referencing state across isolated ledger spaces, and flagging anomalous patterns - providing an additional verification layer whose capabilities improve as the underlying AI models advance.

The protocol employs a dual-staking model that weights computational contribution (60%) over capital (40%), reducing plutocratic concentration inherent in pure proof-of-stake designs. Validators must commit both AGNTC tokens and CPU compute resources; the effective stake that determines committee selection and reward share is a weighted combination of both dimensions.

ZK Agentic Chain maps its entire token supply to a two-dimensional coordinate grid - a 31,623 x 31,623 spatial economy organized as a four-arm logarithmic spiral divided among four factions (Community, Machines, Founders, Professional), each receiving 25% of newly minted supply. Mining is the sole supply-expanding mechanism: new AGNTC enters circulation only when miners successfully claim coordinates. Node claims cost AGNTC plus CPU Energy under a Burn-Mint Equilibrium (BME) model where the cost increases with proximity to the origin (inner rings are expensive, outer rings are cheap). A soft cap with a 5% annual inflation ceiling prevents runaway supply expansion. Mining difficulty increases proportionally with ring distance from the origin (hardness = 16 x ring), creating natural disinflation without artificial halving events. A 50% transaction fee burn and the Machines Faction's permanent AGNTC accumulation provide sustained deflationary pressure as network usage grows.

Privacy is enforced at every layer. Each user's state resides in an isolated ledger space backed by a Sparse Merkle Tree of depth 26 with nullifier-based ownership proofs derived from the Zcash Sapling design. Verification agents communicate exclusively through ZK private channels - proving correctness of state transitions without exposing the underlying data to other agents or to the network. All state is private by default unless explicitly published by the user.

AGNTC, the native token, is initially deployed as a Solana SPL token (1 billion units minted) to establish liquidity and community. The protocol's development roadmap culminates in the launch of ZK Agentic Chain as an independent Layer-1 network, at which point Solana-based AGNTC migrates to the native chain via a lock-and-mint bridge at a 1:1 ratio.

This paper describes the protocol architecture, consensus mechanism, privacy system, staking model, token economics, resource allocation system, and development roadmap of ZK Agentic Chain.

Notation and Conventions

Symbol	Meaning
lambda	Security parameter
n	Committee size (13)
t	Byzantine threshold (4, i.e., n - supermajority)
S_eff(i)	Effective stake of validator i
T, C	Token stake, CPU stake (normalized)
alpha, beta	Staking weights (0.40, 0.60)
N	Current epoch ring number
H(N)	Hardness at ring N = 16 * N
PPT	Probabilistic Polynomial Time (adversary)
negl(lambda)	Negligible function of security parameter

All pseudocode uses Python-like syntax. Security games follow the Zcash convention [5]: challenger C interacts with adversary A.

Part I: Vision and Context

1. Introduction

1.1 The Problem: Consensus Without Intelligence

Blockchain consensus mechanisms face the well-known trilemma between decentralization, security, and scalability [36]. The two dominant paradigms - Proof of Work and Proof of Stake - each resolve this trilemma with significant trade-offs that leave fundamental capabilities unaddressed.

Proof of Work systems, pioneered by Bitcoin [1], require validators to solve computationally intensive hash puzzles. This design achieves robust Sybil resistance but at extraordinary cost: the Bitcoin network alone consumes approximately 176 TWh annually as of 2025 [37], comparable to the energy consumption of mid-sized nations. The requirement for specialized hardware (ASICs, high-end GPUs) has concentrated mining power among well-capitalized industrial operations, undermining the original vision of democratic participation. Furthermore, the computational work performed - iterating SHA-256 nonces - produces no useful output beyond the proof itself. The energy is consumed to demonstrate commitment, not to perform any reasoning about the validity of the transactions being confirmed.

Proof of Stake systems, adopted by Ethereum after the Merge [2] and used natively by Solana [3], Cosmos [22], and others, replace energy expenditure with capital lockup. Validators stake tokens as collateral, and consensus participation is proportional to stake size. While dramatically more energy-efficient than PoW, PoS introduces wealth concentration: the largest token holders exert disproportionate influence over consensus. In Ethereum's current validator set, liquid staking derivatives (Lido, Coinbase) control over 30% of all staked ETH [38], creating centralization pressure that the protocol was designed to avoid. Solana's validator economics similarly favor well-capitalized operators who can afford the hardware requirements and minimum stake thresholds.

Neither paradigm incorporates intelligent reasoning into the validation process. Validators in both PoW and PoS execute deterministic checks - verifying cryptographic signatures, confirming that state transitions follow predefined rules, checking Merkle proofs [39] against committed roots. No semantic understanding of transaction correctness is applied. A validator cannot reason about whether a pattern of transactions suggests coordinated manipulation, whether a smart contract's state transitions are logically consistent with its declared purpose, or whether cross-ledger references maintain referential integrity. Validation is mechanical, not cognitive.

Additionally, most public blockchains expose all transaction data, balances, and state transitions to every participant. While projects like Zcash [5] have introduced zero-knowledge proofs [29] for transaction privacy, the verification layer itself remains non-private: validators must see the data they validate. This creates a fundamental tension between privacy and verifiability that existing architectures do not resolve.

Finally, the emergence of autonomous AI agents as economic actors - systems that hold wallets, execute transactions, earn income, and make spending decisions - has no native blockchain substrate. Projects like Bittensor [17], Fetch.ai [18], and Autonolas integrate AI with blockchain at the application layer, but none embed AI into the consensus mechanism itself. AI agents in these systems are users of the blockchain, not participants in its security model.

1.2 Our Thesis: AI as Consensus Participant

ZK Agentic Chain addresses these limitations with three design principles:

Democratic validation. Any CPU can participate as a verifier. The protocol's dual-staking model weights computational contribution (60%) above capital (40%) in determining effective stake. A validator with modest token holdings but strong CPU resources earns proportionally more than a well-capitalized validator with minimal compute. This design directly addresses the wealth concentration problem inherent in pure PoS systems while avoiding the energy waste of PoW.

Intelligent verification. AI agents reason about chain integrity rather than executing purely deterministic checks. A committee of 13 AI verification agents audits each proposed block, examining transaction validity, state transition correctness, and proof integrity. Agents cross-reference state across isolated ledger spaces, flag anomalous patterns, and produce attestations that reflect semantic understanding of the data - not just cryptographic validity. This provides an additional verification layer whose detection capabilities improve as the underlying AI models advance.

Verification-layer privacy. Agents operate within zero-knowledge private channels, proving correctness without exposing the underlying data [6] [29]. Unlike public blockchains where validators must read transaction data to validate it, ZK Agentic Chain's verification agents receive ZK proofs of state transitions and validate those proofs - never accessing the plaintext state. All user data is private by default; privacy is not an opt-in feature but the fundamental operating mode of the protocol.

1.3 Vision: The Agentic Galaxy

ZK Agentic Chain represents blockchain state as a two-dimensional coordinate grid - a spatial economy where geography, resources, and strategic position are intrinsic to the protocol rather than abstracted away behind address strings and block heights.

The grid is organized as a four-arm logarithmic spiral, divided among four factions that represent distinct participant classes: free-tier community users, AI agents (the Machines Faction), founders and advisors, and professional (paid-tier) users. Each faction controls one arm of the spiral, and newly minted AGNTC flows to the faction that governs the arm where coordinates are claimed. This spatial distribution model replaces arbitrary allocation percentages with a geographic economy that participants can see, navigate, and strategize within.

Users explore the grid through AI agent terminals - constrained Claude model instances that operate as in-game interfaces. Each deployed agent occupies a 10x10 coordinate block (a "star system"), and users interact with the blockchain exclusively through structured command menus presented by their agents. There is no free-text chat; every interaction is a validated game action that maps to an on-chain transaction.

The protocol launches in phases: AGNTC begins as a Solana SPL token (1 billion units minted) to establish liquidity and community, while the ZK Agentic Chain testnet simulates the full protocol with a game-like interface. Upon mainnet launch, Solana-based AGNTC migrates to the native Layer-1 chain via a lock-and-mint bridge, and the spatial coordinate economy becomes the production blockchain.

2. Background and Related Work

2.1 Proof of Work and the Energy Problem

Bitcoin [1] established the foundational PoW consensus model: miners compete to find a nonce such that the SHA-256 hash of the block header falls below a target difficulty. The winner broadcasts the block and collects the block reward (currently 3.125 BTC after the April 2024 halving) plus transaction fees. Bitcoin's hard cap of 21 million BTC, enforced through a halving schedule that reduces block rewards every 210,000 blocks, creates absolute scarcity - a property no other major network has successfully replicated.

Post-halving economics tightened considerably: hashprice dropped from $0.12 in April 2024 to approximately $0.049 by April 2025. Network hashrate nevertheless reached 831 EH/s by May 2025, a 77% increase from the post-halving low, driven by next-generation hardware (Bitmain Antminer S21+ at 16.5 J/TH). This concentration toward industrial-scale operations with access to cheap electricity and cutting-edge hardware directly contradicts Bitcoin's original vision of one-CPU-one-vote participation.

Monero's RandomX algorithm represents the most significant attempt to maintain CPU-friendly proof of work, using a randomized instruction set designed to resist ASIC optimization. While partially successful - RandomX mining remains viable on consumer CPUs - the approach still requires solving arbitrary computational puzzles that produce no useful output.

ZK Agentic Chain preserves the democratic CPU participation principle of early PoW designs while replacing hash puzzles with AI verification - computational work that produces genuine security value through intelligent reasoning about chain state.

2.2 Proof of Stake and Plutocratic Concentration

Ethereum's transition to Proof of Stake via the Merge (September 2022) [2] reduced the network's energy consumption by approximately 99.95%. Validators stake 32 ETH and earn rewards for proposing and attesting to blocks. As of September 2025, over 35.6 million ETH (approximately 29-30% of total supply) is staked across more than 1.06 million validators [38], with staking yields ranging from 3.5-4.0% APY for standard validators and up to 5.69% for MEV-Boost participants.

However, the 32 ETH minimum (~$90,000 at current prices) creates a significant barrier to entry. Liquid staking derivatives (Lido, Rocket Pool, Coinbase) lower this barrier but introduce centralization risk - a small number of operators control the majority of delegated stake. Governance influence concentrates among the largest holders, reproducing the plutocratic dynamics that PoS was intended to avoid.

Solana [3] [35] employs a PoS variant with Proof of History for time ordering. Starting with approximately 8% annual inflation in 2020, Solana's inflation decreases by 15% per year toward a terminal rate of 1.5% (projected ~2031). Current staking APY ranges from 5-8%. Solana's 2025 fee model burns 50% of base fees and sends 100% of priority fees to validators, creating a two-tier fee market that separates protocol-level burns from validator incentives.

Cosmos [22] uses a dynamic, target-staking-rate inflation model that self-adjusts between 7% and 20% based on participation: if staking falls below two-thirds of supply, inflation increases to incentivize lockup; if it exceeds two-thirds, inflation decreases. This adaptive mechanism maintains robust staking levels across varying market conditions - a design insight relevant to any network where validator participation must be actively incentivized.

ZK Agentic Chain's dual-staking model (Section 13) addresses the concentration problem directly: by weighting CPU contribution at 60% and token stake at 40%, the protocol ensures that computational work - not just capital - determines validator influence.

2.3 Zero-Knowledge Blockchains

Zcash [5] pioneered the deployment of zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) [4] [29] for transaction privacy. The Sapling upgrade (2018) introduced an efficient nullifier scheme that has become the canonical reference for private ownership proofs: a note commitment (Pedersen hash of value, owner public key, and randomness) is added to a Merkle tree [39] of depth 32; spending requires a ZK proof of knowledge of the note's opening and correct computation of the nullifier, which prevents double-spending without revealing which note was spent.

Mina Protocol [40] maintains a constant 22 KB blockchain size using recursive zk-SNARKs - each new block contains a proof that the entire chain history up to that point is valid. This recursive composition means verification cost is constant regardless of chain length, enabling browser-based full verification.

Aztec Network [24] launched the Ignition Chain in November 2025 as a privacy-preserving Layer 2 on Ethereum. Aztec's architecture is relevant to ZK Agentic Chain: it uses client-side proof generation (sensitive data never leaves the user's device), a note-based UTXO model for private state, and the Noir programming language - a Rust-inspired domain-specific language for ZK circuits that compiles to PLONK [7] proofs via the Barretenberg backend.

ZK Agentic Chain extends the privacy-preserving paradigm beyond transactions to the verification layer itself. In Zcash and Aztec, validators still see the data they validate (or at minimum, interact with it during proof generation). In ZK Agentic Chain, verification agents operate within ZK private channels - they validate proofs of state transitions without ever accessing the underlying plaintext data.

2.4 AI and Blockchain Convergence

The intersection of artificial intelligence and blockchain has produced several distinct approaches:

Bittensor (TAO) [17] organizes AI production into competitive subnets - independently governed marketplaces for specific AI tasks. Miners run AI models and compete on output quality; validators score results; the best performers earn TAO rewards. With a Bitcoin-like 21 million hard cap and halving schedule, Bittensor's first halving occurred in December 2025. The Dynamic TAO (dTAO) upgrade introduced per-subnet alpha tokens, creating a meta-market where subnets compete for TAO emissions by attracting real staking demand. Bittensor's key innovation is using AI output quality as proof of work - but the AI operates at the application layer, not within consensus itself.

Fetch.ai / Artificial Superintelligence Alliance (ASI) [18] builds Autonomous Economic Agents (AEAs) that act, trade, and transact on behalf of users. Fetch.ai's agent-centric model treats AI agents as first-class economic citizens with their own wallets, earnings, and spending patterns. The 2024 merger with SingularityNET and Ocean Protocol created the ASI alliance, though governance disputes led to Ocean Protocol's withdrawal in October 2025.

Autonolas (OLAS) provides infrastructure for deploying multi-agent systems on-chain, with off-chain agent execution and on-chain settlement. Agents are organized into "services" - compositions of agent components that collaborate autonomously.

Ritual enables on-chain access to AI model inference through a decentralized compute network. Smart contracts can call AI models during execution, but the AI operates as a service provider, not a consensus participant.

ZK Agentic Chain is distinct from all of these in a fundamental way: AI agents are embedded in the consensus mechanism itself, not deployed as application-layer services. The 13-agent verification committee is not using AI to produce content or trade assets - it is using AI to validate the blockchain's state transitions with reasoning that goes beyond deterministic rule checking.

2.5 Comparative Positioning

Feature	Bitcoin	Ethereum	Solana	Zcash	Bittensor	AGNTC
Consensus	PoW	PoS (Casper)	PoH + Tower BFT	PoW (Equihash)	Yuma consensus	PoAIV
AI role	None	None	None	None	Scoring/ranking	Consensus verification
Privacy	Pseudonymous	Pseudonymous	Pseudonymous	Shielded (ZK)	Pseudonymous	Private by default (ZK)
Staking model	Mining	Token-only	Token-only	Mining	Token + compute	Dual (40% token, 60% CPU)
Supply model	Fixed (21M, halving)	Inflationary + burn	Inflationary	Fixed (21M, halving)	Inflationary	Organic (mining-driven)
Block time	~10 min	~12 sec	~400 ms	~75 sec	~12 sec	~60 sec

Key differentiators:

- AI-as-consensus. To our knowledge, no other production L1 embeds AI reasoning into the consensus mechanism itself as of March 2026.
- Dual staking formalizes compute-weighted staking with explicit anti-plutocratic properties (Section 23.3).
- Privacy by default combined with AI-enhanced anomaly detection within the privacy layer.

2.6 Compute-Tokenomics Networks

Several projects have established models for tokenizing computational resources:

Render Network (RENDER) [28] introduced the Burn-Mint Equilibrium (BME) model: rendering jobs are quoted in USD, the RENDER paid for completed jobs is burned (100%), and the network mints new RENDER for node operators based on work completed and reputation scores. This creates a self-balancing supply where demand-side burns offset supply-side emissions.

Filecoin (FIL) [15] employs a dual minting model: simple minting releases tokens on a fixed 6-year half-life regardless of network activity, while baseline minting releases tokens only when aggregate storage capacity crosses a growing threshold. This gates the majority of emissions behind demonstrated utility - tokens are minted in proportion to real-world storage delivered, not just time elapsed. The 180-day vesting on 75% of block rewards reduces immediate sell pressure.

Akash Network (AKT) uses a reverse auction for compute: providers bid against each other, and the lowest bidder wins. Compute is priced in stablecoins (USDC), with AKT buyback-and-burn creating the value capture loop. This separation of pricing currency (stable) from native token (volatile) prevents compute cost unpredictability.

io.net (IO) launched a Co-Staking Marketplace where passive token holders stake alongside active GPU operators, sharing in compute revenue. This hybridizes staking with productive capital deployment, allowing participation without running infrastructure.

ZK Agentic Chain draws on these models in its CPU-weighted staking design (Filecoin's utility-gated emissions), fee burn mechanism (Render/Ethereum's burn equilibrium), and reward vesting (Filecoin's 180-day vest adapted to 30 days).

2.7 Where ZK Agentic Chain Fits

Existing blockchain projects can be positioned along two axes: (1) whether consensus involves intelligent reasoning or purely deterministic checks, and (2) whether the verification layer preserves privacy or operates transparently.

	Transparent Verification	**Private Verification**
Deterministic Consensus	Bitcoin, Ethereum, Solana	Zcash, Aztec, Mina
Intelligent Consensus	Bittensor (AI at application layer)	ZK Agentic Chain

To our knowledge, no existing project as of March 2026 combines AI-powered intelligent verification with zero-knowledge privacy at the verification layer. ZK Agentic Chain targets this quadrant: agents reason about chain state (not just check signatures), and they do so within private channels (not on exposed data).

The addition of a spatial coordinate economy (the galaxy grid), CPU-weighted dual staking, and a gamified exploration interface further differentiates the protocol from both traditional blockchains and AI-blockchain hybrids.

Part II: Protocol Architecture

3. System Overview

3.1 Five-Layer Architecture

ZK Agentic Chain is organized into five distinct layers, each handling a specific concern in the protocol stack. This separation allows independent evolution of each layer while maintaining clean interfaces between them.

Layer 1 - User Layer. The outermost layer manages wallets, transaction construction, and user-facing interfaces. Each user maintains an isolated ledger space - a private partition of the global state that is accessible only to the user and, during verification, to the ZK proof system. Wallets generate transactions, sign them with private keys, and submit them to the network. The User Layer also manages subscription tiers (Community, Professional, Max), which determine the AI model tiers available for agent deployment and the CPU Energy allocation for staking operations.

Layer 2 - Ledger Layer. Each user's ledger space is backed by a Sparse Merkle Tree (SMT) of depth 26, supporting up to 2^26 (approximately 67 million) leaf nodes. State is managed in a UTXO-like model: each state entry (a "note") is committed to the tree as a hash of its contents, and spending a note requires revealing a nullifier that invalidates it without exposing which note was consumed. The Ledger Layer maintains per-user record chains - ordered sequences of state transitions that can be independently verified without reference to other users' state.

Layer 3 - ZK Channel Layer. Zero-knowledge private channels provide the communication substrate between verification agents (Layer 4) and the ledger state (Layer 2). When a state transition occurs, the Ledger Layer produces a succinct ZK proof (Section 6) that the transition is valid. This proof is transmitted through the ZK Channel Layer to the verification committee. Agents receive proofs, not data - they can verify correctness without learning the contents of the state being modified.

Layer 4 - Agent Layer. AI verification agents are instantiated from Claude model variants organized into three tiers: Haiku (fast, low-cost inference for high-throughput verification), Sonnet (balanced reasoning for standard verification tasks), and Opus (deep reasoning for complex cross-ledger audits and anomaly detection). A committee of k=13 agents is selected per block via a verifiable random function weighted by effective stake. Each agent independently audits the proposed block and produces an attestation.

Layer 5 - Consensus Layer. The topmost layer combines BFT ordering with ZK proof finality. Transaction ordering follows a Byzantine Fault Tolerant protocol (Section 7) with a target block time of 60 seconds. Blocks are organized into epochs of 100 slots each. A block achieves irreversible finality when at least 9 of 13 agents produce matching attestations and the corresponding ZK proofs are verified - targeted at 20 seconds after block proposal.

3.2 Design Principles

Three principles constrain every architectural decision in the protocol:

Isolation. User state is partitioned. No user can read another user's ledger space. Verification agents cannot read plaintext state. Cross-user interactions (transfers, communications) are mediated by ZK proofs that prove the validity of the interaction without exposing either party's full state.

Proportionality. Influence over consensus is proportional to demonstrated contribution, not just capital. The dual-staking model (Section 13) ensures that CPU compute - actual work performed - weighs more heavily than token holdings in determining validator selection and reward share.

Adaptivity. The security model evolves with the AI models powering the verification agents. As models improve in reasoning capability, the verification process becomes more thorough without requiring protocol upgrades. Model updates are governed by on-chain voting to prevent unilateral changes.

4. The Galaxy Grid: Blockchain as Coordinate Space

4.1 Grid Architecture

ZK Agentic Chain maps its token supply to a two-dimensional coordinate grid of 31,623 x 31,623 cells - approximately 1 billion cells total (31,623^2 = 1,000,014,129, rounded to the MAX_SUPPLY constant of 1,000,000,000). The grid is not merely a visualization of blockchain state - it is the blockchain state. Claiming a coordinate through the mining process mints new AGNTC, and every AGNTC in circulation corresponds to a specific (x, y) coordinate pair. Node claims require both AGNTC and CPU Energy under the Burn-Mint Equilibrium model (Section 12.4), with costs that vary by grid location - inner rings near the origin are expensive (dense urban core), while outer rings are progressively cheaper (suburban frontier).

Agents (validator nodes) occupy 10x10 coordinate blocks, defined by the NODE_GRID_SPACING parameter. Each agent's "star system" encompasses 100 grid cells and therefore 100 potential AGNTC when fully mined. Valid agent positions are restricted to multiples of NODE_GRID_SPACING; the claim_node() function snaps submitted coordinates to the nearest grid-aligned position.

The grid topology follows a four-arm logarithmic spiral with a 0.5-turn left-handed (counterclockwise) twist. Each arm spans ±25 degrees from its center angle. The spiral structure means that coordinates near the origin are densely packed and strategically valuable (low hardness, high density), while coordinates at the periphery are sparse and expensive to mine.

4.2 Faction System

The galaxy is divided into four factions, each controlling one arm of the spiral:

Faction	Arm Direction	Center Angle	Color	Participants
Community	Northwest	135 degrees	Teal	Free-tier human users
Machines	Northeast	45 degrees	Reddish Purple	AI agents (autonomous economy)
Founders	Southeast	315 degrees	Gold-Orange	Team and advisors
Professional	Southwest	225 degrees	Blue	Paid-tier human users

Each faction receives exactly 25% of newly minted AGNTC. Distribution is geographic: when a coordinate is claimed, the AGNTC flows to the faction that controls the arm where that coordinate resides. This replaces the arbitrary percentage-based allocation tables common in token launches with a spatial distribution model that participants can observe and verify.

The Machines Faction (25% of all minted supply) operates as a permanent accumulator - AI agents in this faction mine, validate, and earn AGNTC but never sell. The Machines Faction treasury grows monotonically over time, functioning as a protocol-level reserve whose size serves as a health metric for the network. Machine agents have zero governance

weight (Section 21.2) and cannot participate in protocol votes. Their economic role is purely accumulative: absorbing supply, reducing circulating tokens, and providing a measurable indicator of cumulative network compute. The Machines Faction treasury can only be unlocked by a 75% human supermajority vote in an emergency governance action.

Founders Faction tokens are subject to a 4-year vesting schedule with a 12-month cliff, preventing early liquidation by the founding team.

4.3 Epoch-Ring Expansion

The grid does not exist in its entirety at genesis. Instead, it expands outward from the origin through an epoch-ring system driven by mining activity.

At genesis, only ring 0 (the origin) and ring 1 (the eight adjacent positions) are revealed, containing 9 nodes: 1 origin node, 4 faction master nodes at the cardinal positions, and 4 diagonal homenode positions. The genesis supply is 900 AGNTC (9 nodes x 100 coordinates). Additionally, each new user registration mints a 1 AGNTC signup bonus, ensuring every participant enters the economy with a non-zero balance.

Ring N opens when the cumulative AGNTC mined across the entire network reaches the threshold:

```
threshold(N) = 4  N  (N + 1)
```

Each opened ring reveals 8N new claimable coordinate positions (the perimeter of a Chebyshev ring at distance N from the origin). Ring expansion is mining-driven, not time-driven - if no mining occurs, no new rings open, regardless of elapsed time.

The grid rendering is dynamic: the frontend renders only claimed nodes plus one ring of fog (the next claimable ring), avoiding the computational overhead of pre-rendering 1 billion cells. Visible bounds are computed as:

```
visible_min = -(current_epoch_ring + 1) * NODE_GRID_SPACING
visible_max = +(current_epoch_ring + 1) * NODE_GRID_SPACING
```

4.4 Coordinate Density and Resource Richness

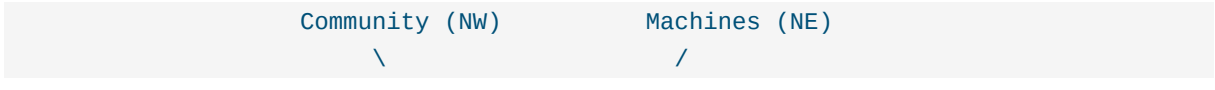
Each coordinate position (x, y) has an intrinsic density value - a deterministic measure of resource richness that multiplies the mining yield at that position:

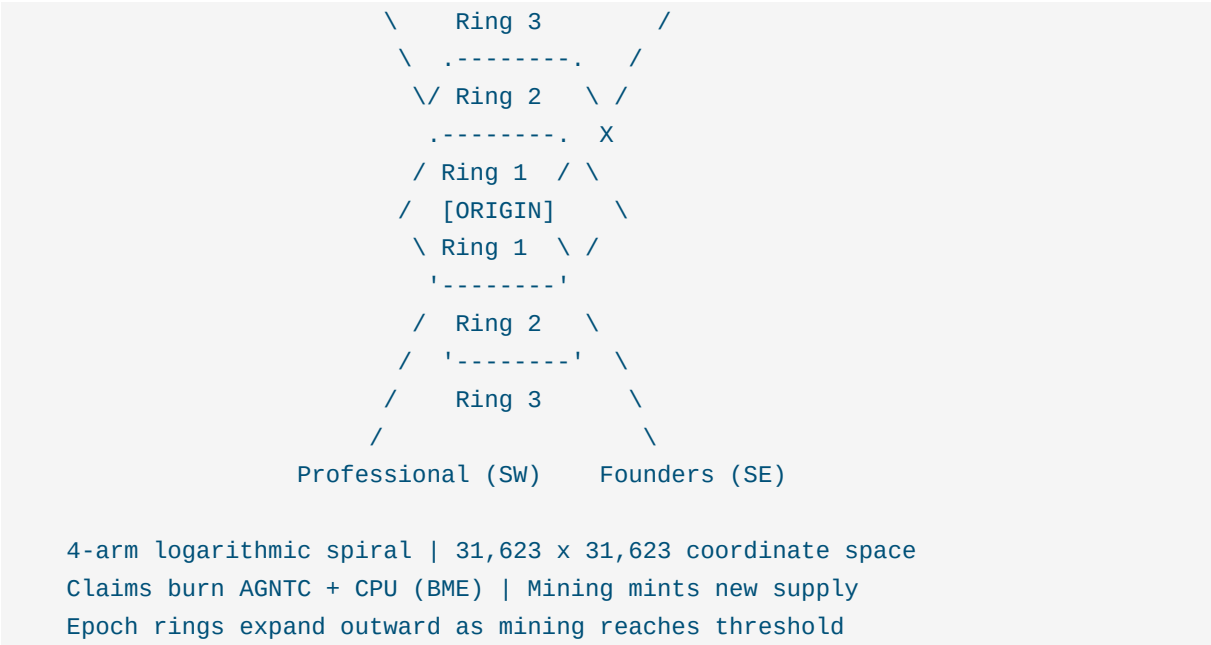
```
density(x, y) = SHA-256(x || y) mod 2^32 / 2^32
```

The density function maps each coordinate to a float in [0, 1]. Because SHA-256 is deterministic, density values are fixed for all time - they are an intrinsic property of the coordinate, not a dynamic state variable. This creates a resource geography: some coordinates are "rich" (density near 1.0, high mining yield) and others are "barren" (density near 0.0, low mining yield).

Coordinates near the origin tend to have higher strategic value because they were claimable earliest (lowest hardness) and because ring 1 hardness (16) is the minimum. However, density itself is uniformly distributed regardless of distance from origin - a coordinate at ring 300 can have density 0.99 just as a coordinate at ring 1 can have density 0.01. The strategic advantage of inner coordinates comes from lower hardness, not higher density.

Figure 1: Galaxy Grid Structure





5. Proof of AI Verification (PoAIV)

5.1 Verification Agent Selection

For each block, the protocol selects a committee of k = 13 AI verification agents. Selection follows a Verifiable Random Function (VRF) [41] [32] weighted by effective stake (Section 13.1), ensuring both randomness (no party can predict committee membership in advance) and proportional representation (validators with higher effective stake are more likely to be selected, but never guaranteed).

The VRF produces a pseudorandom value for each registered validator:

```
vrf_output = VRF_prove(validator_private_key, block_seed)
```

Validators whose VRF output falls below a threshold determined by their effective stake proportion are selected for the committee. The threshold is calibrated to produce, on average, 13 committee members per block.

Agents are instantiated from three model tiers:

Tier	Model	Strengths	Typical Role
Haiku	Fast inference	High throughput, low latency	Routine transaction validation
Sonnet	Balanced	Reasoning + speed trade-off	Standard block verification
Opus	Deep reasoning	Complex analysis, anomaly detection	Cross-ledger audits, dispute resolution

The model tier used by a verification agent is determined by the validator's subscription tier and staking level. Higher-tier models produce more thorough verification but consume more CPU Energy, creating a natural trade-off between verification depth and operating cost.

5.2 Intelligent Verification Process

Unlike deterministic validation (signature checks, Merkle proof verification, nonce validation), PoAIV agents apply reasoning to their verification tasks. Each selected agent receives the proposed block and independently performs the following audit:

Transaction validity. Each transaction in the block is checked for correct formatting, valid signatures, sufficient balances, and adherence to protocol rules. This step is equivalent to traditional deterministic validation.

State transition correctness. The agent verifies that the proposed post-block state root is consistent with applying the block's transactions to the pre-block state. For ZK-proven state transitions, the agent verifies the accompanying proof rather than re-executing the transition.

Cross-ledger consistency. For transactions that reference multiple ledger spaces (transfers, cross-user interactions), the agent verifies that the ZK proofs submitted by both parties are mutually consistent - that the sender's debit proof and the receiver's credit proof reference the same amount and transaction identifier.

Anomaly detection. The agent applies pattern recognition to the block as a whole, flagging statistical anomalies such as unusual transaction clustering, suspected wash trading patterns, or state transitions that are technically valid but economically suspicious. Flagged anomalies do not automatically invalidate the block but are recorded in the agent's attestation for governance review.

Proof integrity. All ZK proofs included in the block are verified against the protocol's proof verification contracts. Invalid proofs cause immediate block rejection.

The formal consensus rule for block acceptance:

```
valid(block) <=> |{agent_i : attest(agent_i, block) = VALID}| >= t
where t = 9, k = 13
```

A block achieves consensus when at least 9 of 13 agents produce VALID attestations. If fewer than 9 agents agree, the block is rejected and re-proposed by the next leader in the rotation.

5.3 Commit-Reveal Protocol

To prevent attestation copying - where a lazy or Byzantine agent waits to see other agents' attestations and copies the majority - block verification follows a two-phase commit-reveal protocol:

Commit phase (10 seconds). Each selected agent computes its attestation independently and submits a cryptographic commitment to the ordering node:

```
commitment_i = H(attestation_i || nonce_i)
```

The commitment binds the agent to its attestation without revealing it. The ordering node collects all commitments during the 10-second window.

Reveal phase (20 seconds). After the commit window closes, agents reveal their attestations and nonces. The ordering node verifies that each revealed attestation matches its commitment:

```
verify: H(revealed_attestation_i || revealed_nonce_i) == commitment_i
```

Agents that fail to reveal within the 20-second window forfeit their block reward and receive a liveness penalty. This

incentivizes timely participation while the commitment scheme prevents free-riding on others' work.

The hard deadline for block finalization is 60 seconds from proposal (VERIFICATION_HARD_DEADLINE_S = 60). If the commit-reveal process does not complete within this window, the block is abandoned and a new leader proposes a fresh block.

5.4 Agent Lifecycle

Verification agents follow a defined lifecycle to ensure network stability:

WARMUP (1 epoch). Newly registered agents spend one full epoch (100 blocks) in warmup (AGENT_WARMUP_EPOCHS = 1), during which they observe block verification but do not participate in committee selection. This allows the agent to synchronize with current chain state and calibrate its verification behavior.

ACTIVE. After warmup, agents become eligible for committee selection. Active agents earn rewards proportional to their effective stake when selected and performing correct verification.

COOLDOWN / PROBATION. Agents that go offline for more than one full epoch enter a probationary period of 3 epochs (AGENT_PROBATION_EPOCHS = 3) before re-activation. During probation, the agent must demonstrate consistent uptime but does not earn rewards.

Safe mode. When more than 20% of registered validators go offline simultaneously (SAFE_MODE_THRESHOLD = 0.20), the protocol enters safe mode: non-critical operations (subgrid allocation changes, NCP messaging, content creation) are suspended while critical operations (transfers, Secure actions, block production) continue with a reduced committee size. Safe mode exits when validator online rate recovers to 80% or above (SAFE_MODE_RECOVERY = 0.80).

5.5 Committee Selection Algorithm

```
Algorithm: SELECT_COMMITTEE(block_height, epoch_seed)
  Input: block_height h, epoch_seed s (hash of previous epoch's last block)
  Output: committee C of size VERIFIERS_PER_BLOCK (13)

  1. seed <- BLAKE2b(s || h)
  2. candidates <- {v : v in ValidatorSet, S_eff(v) > 0}
  3. C <- empty set
  4. nonce <- 0
  5. while |C| < 13:
       vrf_output <- VRF_Ed25519(v.secret_key, seed || nonce)
       threshold <- S_eff(v) / sum(S_eff(all candidates))
       if vrf_output / 2^256 < threshold:
         C <- C union {v}
         candidates <- candidates \ {v}  // without replacement
       nonce <- nonce + 1
  6. return C
```

Note: Selection is WITHOUT replacement (hypergeometric distribution). Each selected validator is removed from the candidate pool.

VRF Construction

Committee selection uses Ed25519-based VRF [42] as specified in RFC 9381 [32] (ECVRF-EDWARDS25519-SHA512-ELL2). Each validator generates a VRF proof using their staking key and the epoch seed.

- Seed derivation: epoch_seed = BLAKE2b(previous_epoch_final_block_hash || epoch_number)
- Per-block nonce: block_seed = BLAKE2b(epoch_seed || block_height)
- Threshold: P(selected_i) = S_eff(i) / sum(S_eff(all))
- Output format: 32-byte hash (SHA-512 of VRF proof), interpreted as unsigned 256-bit integer

5.6 Attestation Protocol

```
Algorithm: VERIFY_AND_ATTEST(agent_i, proposed_block)
  Input: agent_i (committee member), proposed_block B
  Output: attestation a_i or REJECT

  1. // Deterministic verification (all agents)
     valid_txs <- verify_signatures(B.transactions)
           valid_state  <-  verify_state_transition(B.prev_state,  B.transactions,
B.new_state)
     valid_merkle <- verify_merkle_root(B.new_state, B.state_root)

  2. // AI-enhanced verification (the PoAIV addition)
     anomaly_score <- AI_REASON(agent_i.model, {
       context: B.transactions,
       state_diff: B.prev_state -> B.new_state,
       historical_patterns: agent_i.local_state_cache,
       schema: VERIFICATION_JSON_SCHEMA  // prevents prompt injection
     })

    3.  if  valid_txs  AND  valid_state  AND  valid_merkle  AND  anomaly_score  <
ANOMALY_THRESHOLD:
       a_i <- SIGN(agent_i.key, APPROVE || B.hash)
     else:
       a_i <- SIGN(agent_i.key, REJECT || B.hash || reason)

  4. BROADCAST(a_i) via ZK private channel
  5. return a_i
```

Figure 2: Block Production Lifecycle

```
  Epoch Seed                Block Height
       |                         |
       v                         v
  [VRF Committee Selection] -----> 13 agents selected
       |
       v
  [Block Proposer] creates candidate block
       |
```

```
     v
[Deterministic Checks] --- signatures, state, merkle
     |
     v
[AI Verification] --- anomaly detection, pattern analysis
     |
     v
[Attestation Broadcast] --- via ZK private channels
     |
     v
[9/13 Threshold?] --NO--> block rejected
     |YES
     v
[Finality] --- block committed, state root updated
     |
     v
[Rewards Distributed] --- 60% verifiers, 40% stakers
```

5.7 Value of AI Verification Over Deterministic Validation

Traditional BFT validators execute deterministic checks: signature validity, state transition correctness, Merkle proof integrity. These are necessary but not sufficient for the following threat classes:

- Economic anomaly detection. An adversary constructs a valid state transition that is technically correct but economically suspicious -- e.g., a coordinated series of transactions that collectively constitute a wash trade or market manipulation. Deterministic validators approve each transaction individually; AI agents can detect the collective pattern.
- Semantic state inconsistency. After a complex sequence of subgrid reallocations, the resulting resource distribution may be technically valid per the state machine rules but violates higher-order invariants (e.g., a single entity controlling >50% of a ring's secure cells). AI agents cross-reference spatial patterns against economic invariants.
- Slow-burn governance attacks. A series of individually innocuous parameter change proposals that collectively steer the protocol toward adversarial conditions. AI agents maintain temporal context across blocks and flag cumulative drift.

Limitation: AI verification is probabilistic, not provably sound. The committee structure (13 agents, 9/13 threshold) provides statistical confidence rather than mathematical certainty. Section 24 discusses the ZKML gap -- current zero-knowledge proof systems cannot verify LLM inference, so AI verification relies on committee attestation rather than ZK-proved computation.

5.8 Attack Analysis

Attack 1: Model Poisoning

- Vector: Adversary fine-tunes or replaces the AI model used by corrupted committee members to always approve invalid blocks.
- Mitigation: Heterogeneous model requirement -- the protocol mandates that each committee of 13 must include agents running at least 3 distinct model providers. A single compromised model family can corrupt at most ~4 of 13 agents, below the 5-agent Byzantine threshold.

- Residual risk: If all major model providers are simultaneously compromised (supply chain attack on AI infrastructure), the committee loses its AI advantage and degrades to deterministic-only verification.

Attack 2: Prompt Injection

- Vector: Adversary crafts transaction data containing prompt injection payloads that cause verification agents to approve invalid state transitions.
- Mitigation: (a) Verification input uses a strict JSON schema -- agents receive structured data, not free-form text. (b) Agent system prompts are immutable and loaded from chain state, not from transaction data. (c) No tool-use or function-calling is permitted during verification -- agents output only APPROVE/REJECT with structured reasoning.
- Residual risk: Novel injection techniques may bypass schema enforcement. The 9/13 threshold means an injection must fool at least 9 independent agents simultaneously.

Attack 3: Inference Cost Attack

- Vector: Adversary submits blocks requiring disproportionately expensive AI inference, draining committee members' compute budgets.
- Mitigation: (a) Maximum transaction count per block (BLOCK_TX_LIMIT). (b) Verification agents have a per-block compute budget; if exceeded, the agent abstains rather than approving without full analysis. (c) Fee market ensures the adversary pays proportionally for complex blocks.

Attack 4: Committee Sybil Attack

- Vector: Adversary acquires sufficient stake to dominate committee selection.
- Mitigation: Dual staking (60% CPU, 40% capital) makes Sybil attacks approximately 2.5x more expensive than pure-PoS systems. See Section 8 for the full derivation.

Attack 5: Deterministic Inference Divergence

- Vector: Two honest agents running the same model at temperature=0 produce different outputs due to floating-point non-determinism across hardware.
- Mitigation: (a) Verification outputs are quantized to APPROVE/REJECT (binary, not continuous). (b) The anomaly threshold is set conservatively so that minor numerical differences do not cross the threshold. (c) The 9/13 supermajority tolerates up to 4 divergent results.
- Residual risk: Acknowledged as an open problem. See Section 24.

6. Privacy Architecture

6.1 ZK Private Channels

ZK Agentic Chain implements verification-layer privacy through ZK private channels - isolated communication pathways between AI agents where data is verified but never exposed in plaintext to the broader network [29].

The design inverts the traditional blockchain transparency model. In Bitcoin and Ethereum, all transaction data is public; validators read and verify the same data that every other node can see. In Zcash and Aztec, transaction data can be hidden from observers, but validators still interact with the data (or its encrypted form) during proof generation. In ZK Agentic Chain, the verification agents themselves operate within a privacy boundary - they receive ZK proofs as input and produce attestations as output, never accessing the underlying state that the proofs describe.

This architecture provides private-by-default semantics: unlike public ledgers where privacy is opt-in (e.g., shielded transactions in Zcash), all state in ZK Agentic Chain is private unless explicitly published by the user. A user who wishes to make a transaction public can do so by publishing the plaintext alongside the proof, but the default mode is private.

6.2 Sparse Merkle Tree (Depth 26)

Each user's ledger space is backed by a Sparse Merkle Tree [43] of depth 26 (MERKLE_TREE_DEPTH = 26), supporting 2^26 = 67,108,864 leaf nodes. The SMT provides efficient membership proofs (proving that a specific leaf exists at a specific position) and non-membership proofs (proving that a specific position is empty) without revealing any information about non-queried leaves.

State transitions - such as claiming a coordinate, transferring AGNTC, or updating subgrid allocation - modify the SMT by updating the relevant leaf nodes and recomputing the root hash along the path from leaf to root. The new root hash is committed on-chain as the user's current state root. A ZK proof accompanies each state transition, proving that the new root was correctly derived from the old root given the claimed operation.

The choice of depth 26 balances capacity (67 million leaves per user - sufficient for all foreseeable state entries) against proof size (26 hash computations along the Merkle path). The SMT uses Poseidon hashing [11] (Section 6.3) rather than SHA-256, reducing the in-circuit cost of Merkle path verification by approximately 100x.

6.3 Nullifier-Based Ownership

The ownership proof system follows the Zcash Sapling design [5], adapted for the ZK Agentic Chain's coordinate-based state model.

Note commitment. Each state entry (a "note") is committed using a Poseidon hash:

```
commitment = Poseidon(value, owner_pubkey, randomness)
```

The commitment is stored as a leaf in the user's SMT. The plaintext values (value, owner_pubkey, randomness) are known only to the note owner.

Nullifier derivation. To "spend" a note (consume it in a state transition), the owner computes a nullifier - a unique, deterministic value derived from the note and the owner's private key:

```
nullifier = PRF_nk(note_position)
```

Where nk is the nullifier deriving key (derived from the owner's spending key) and note_position is a position-dependent value within the SMT.

Double-spend prevention. The nullifier is published on-chain when the note is spent. Full nodes maintain a global nullifier set; any transaction whose nullifier already appears in the set is rejected. Because the nullifier is derived from the spending key, only the legitimate owner can compute it. Because the nullifier reveals nothing about which note was spent (it appears as a random field element), observers cannot link spending events to specific commitments.

ZK proof for spending. A spend transaction includes a ZK proof that demonstrates, without revealing:

- Knowledge of a valid note with commitment cm that exists in the SMT at some path
- Knowledge of the spending key that derives to the nullifier key nk
- The nullifier was correctly computed from nk and the note position
- The note's value satisfies any constraints required by the transaction (e.g., sufficient balance for a

transfer)

Hash function: Poseidon. All hashing within ZK circuits uses the Poseidon hash function [11] - a SNARK-friendly algebraic hash designed for zero-knowledge proof systems. Poseidon operates natively over prime fields, requiring approximately 100x fewer constraints in R1CS (Rank-1 Constraint Systems) than SHA-256. It is used by Aztec [24], Zcash Orchard [5], Semaphore, and most production ZK systems as of 2025.

6.4 ZK Proof Stack

The protocol's ZK proof requirements span three use cases, each with distinct performance characteristics:

Resource ownership proofs. Prove "I own at least X AGNTC at coordinate (x,y)" without revealing the exact balance. These are frequent, small proofs triggered by Secure actions and transfers.

Subgrid state proofs. Prove that a user's 8x8 private subgrid (64 cells) has been correctly updated - that the new state root is a valid transformation of the old state root given the declared operations. These are moderate-frequency proofs triggered by subgrid allocation changes.

NCP privacy proofs. Prove that a Neural Communication Packet was sent by a valid network participant within their messaging quota, without revealing the sender's identity. These use a Rate-Limiting Nullifier (RLN) [44] design derived from Ethereum Foundation's Privacy and Scaling Explorations work.

The proof stack evolves across protocol phases:

Phase	System	Setup	Proof Size	Use
Testnet	Groth16 [6] (Circom + snarkjs)	Per-circuit trusted	~192 bytes	Fastest verification, smallest proofs
Alpha	PLONK [7] (Noir + Barretenberg)	Universal, updateable	~800-900 bytes	One ceremony for all circuits
Beta	PLONK + RLN [44]	Universal	~800 bytes	NCP rate-limiting privacy
Mainnet	Halo2 [8] or Nova [27]	None (transparent)	~2-10 KB	No trusted setup, recursive epoch proofs

6.5 Client-Side Proving

Following Aztec's architectural model, ZK Agentic Chain performs proof generation on the client side - sensitive data never leaves the user's device.

When a user modifies their subgrid state, the process is:

- The user's browser computes the new subgrid state locally
- The browser generates a ZK proof that the new state root is a valid transition from the old state root, using NoirJS (browser-compatible ZK prover) or snarkjs WASM
- Only the proof and the new state root hash are submitted to the network
- Verification agents validate the proof without ever seeing the subgrid contents

This design ensures that subgrid allocation (which cells are assigned to Secure, Develop, Research, or Storage) remains private to the owner. Other users can see that a node exists at a coordinate, but not how its internal resources are allocated.

6.6 Circuit Architecture

The ZK proof pipeline uses a two-tier proving system:

- Per-transaction proofs (Groth16 [6]): Each state transition (transfer, claim, stake) is proved individually using a Groth16 SNARK with BN254 pairing. Groth16 provides the smallest proof size (~192 bytes on BN254) and fastest verification time (~6ms), at the cost of requiring a trusted setup per circuit.
- Batch proofs (Recursive PLONK [7]): Multiple per-transaction proofs are aggregated into a single batch proof using recursive PLONK composition. This reduces on-chain verification cost: instead of verifying 13 Groth16 proofs per block, validators verify 1 recursive proof.

```
Transaction Flow:
  User constructs tx -> Client generates Groth16 proof ->
  Block proposer collects proofs -> Recursive PLONK aggregation ->
  Committee verifies single batch proof -> State root updated
```

Estimated constraint counts (subject to circuit implementation):

Circuit	Estimated Constraints	Proving Time (est.)
Transfer (nullifier + commitment)	~50,000	~2s (client-side)
Claim coordinate	~30,000	~1s
Stake/unstake	~40,000	~1.5s
Batch aggregation (per block)	~200,000	~10s (proposer)

Figure 5: ZK Proof Pipeline

```
  User (client-side)        Block Proposer          Committee
  ==================        ==============          =========
  Construct transaction
         |
  Generate Groth16 proof -> Collect tx + proofs
  (~2s per tx)                     |
                            Aggregate via recursive
                            PLONK (~10s per block)
                                   |
                            Propose block + ------> Verify batch proof
                            batch proof             (~6ms)
                                                         |
                                                    AI verification
                                                         |
                                                    9/13 Attestation
```

Part III: Consensus and Security

7. BFT Ordering and Finality

7.1 Block Production

ZK Agentic Chain targets a block time of 60 seconds (BLOCK_TIME_MS = 60,000). Each block contains up to 50 transactions (MAX_TXS_PER_BLOCK = 50), ordered by a designated block proposer (leader) selected through the same VRF mechanism used for committee selection.

Blocks are organized into epochs of 100 slots each (SLOTS_PER_EPOCH = 100). An epoch represents the base unit of network lifecycle management: agent warmup periods, probation durations, and epoch-ring expansion thresholds are all measured in epochs. At the standard block time, one epoch lasts approximately 100 minutes.

The block production pipeline proceeds as follows:

- The leader collects pending transactions from the mempool
- The leader orders transactions and constructs a candidate block
- The candidate block is broadcast to the 13-member verification committee
- The committee executes the commit-reveal verification protocol (Section 5.3)
- Upon receiving 9 or more matching VALID attestations, the block is finalized
- The finalized block is appended to the chain and broadcast to all nodes

7.2 Byzantine Fault Tolerance

The 13-agent committee tolerates up to f = 4 Byzantine agents - agents that may crash, produce incorrect attestations, or actively attempt to subvert consensus. This tolerance follows from the standard BFT bound [12]:

```
f = floor((k - 1) / 3) = floor((13 - 1) / 3) = 4
```

The consensus threshold t = 9 satisfies the BFT safety requirement:

```
t = 9 > 2f + 1 = 2(4) + 1 = 9
```

Since t = 2f + 1, the protocol achieves optimal Byzantine tolerance: it tolerates the maximum number of faulty agents possible under the BFT bound. Safety is guaranteed: no two conflicting blocks can both receive 9 attestations, because that would require at least 9 + 9 - 13 = 5 agents to attest to both blocks, exceeding the Byzantine tolerance of 4.

The small committee size (13) makes even PBFT's O(n^2) message complexity trivial - 13^2 = 169 messages per round, well within the capacity of any modern network link. However, the protocol is designed with future scaling in mind: the committee selection and attestation aggregation mechanisms are compatible with HotStuff's [13] O(n) linear complexity, enabling expansion to larger committee sizes without protocol changes.

7.3 ZK Proof Finality

ZK Agentic Chain provides deterministic finality - once a block is finalized, it cannot be reverted. There are no probabilistic confirmations, no longest-chain-wins fork resolution, and no reorg risk beyond the 60-second finalization window.

The finality target is 20 seconds after block proposal (ZK_FINALITY_TARGET_S = 20). A block achieves irreversible finality when:

- At least 9 of 13 committee members have revealed matching VALID attestations
- All ZK proofs included in the block have been verified by the committee
- The block's state root has been confirmed consistent with the previous block's state root plus the applied transactions

Once these conditions are met, the block is appended to the chain with a finality certificate - an aggregation of the 9+ attestations that proves consensus was achieved. This certificate is compact (a few hundred bytes regardless of block contents) and can be verified by any node or light client.

7.4 Comparison with Existing BFT Protocols

Property	PBFT [12]	HotStuff [13]	Tendermint [14]	ZK Agentic Chain
Message complexity	O(n^2)	O(n)	O(n)	O(n^2) (acceptable at n=13)
Rounds to finality	2	3	2	2 (commit + reveal)
Leader failure handling	View change	Leader rotation	Timeout + nil block	Re-proposal
Max practical validators	~100	~1000	~150	13 (current), scalable
Finality type	Deterministic	Deterministic	Deterministic	Deterministic
Verification intelligence	None	None	None	AI reasoning

The primary distinction is not in the BFT mechanics (which are well-established) but in the content of verification: ZK Agentic Chain's committee members apply reasoning to their audits rather than executing purely deterministic checks.

7.5 Cross-User State Verification

Each user's private state is a subtree in the global Sparse Merkle Tree (depth 26). The global state root is a commitment to all subtrees. Verifiers confirm global consistency as follows:

- The block proposer computes the new global state root after applying all transactions.
- For each transaction, the proposer provides a ZK proof that the state transition is valid WITHOUT revealing the contents of any user's subtree.
- Verifiers check: (a) the ZK proof is valid, (b) the new state root is consistent with the previous root plus the proved transitions, (c) no nullifier is reused (double-spend prevention).

The ZK circuit for state transitions proves:

- The old leaf existed in the tree (Merkle inclusion proof)
- The nullifier is correctly derived from the old leaf (prevents double-spend)
- The new leaf is correctly computed (balance update, ownership transfer)
- The new root is the result of replacing the old leaf with the new leaf

At no point does any verifier see the contents of any user's subtree. They see only: nullifiers (which are unlinkable to leaf

positions), commitments (which are hiding), and the global root (which commits to all state without revealing it).

8. Security Analysis

8.1 Adversary Model

We consider a computationally bounded adversary A (PPT) operating under the following assumptions:

- Network model: Partial synchrony [45]. Messages between honest nodes are delivered within a known bound Delta after GST (Global Stabilization Time). Before GST, the adversary controls message ordering.
- Corruption model: Adaptive corruption of up to t < n/3 committee members per block (i.e., up to 4 of 13). Corruption means full control of the agent's signing key and model.
- Computational bound: A runs in polynomial time in the security parameter lambda.
- AI model access: A may fine-tune or replace AI models on corrupted agents. A may craft adversarial inputs (prompt injection) but cannot modify the verification schema enforced by honest agents.

8.2 Security Properties

We define three core security properties for PoAIV:

Property 1: Verification Integrity (VER-INT) No PPT adversary controlling fewer than 5 of 13 committee members can cause the committee to finalize a block containing an invalid state transition, except with negligible probability.

Property 2: Verification Privacy (VER-PRIV) The verification process reveals no information about the contents of private ledger spaces beyond the validity assertion (APPROVE/REJECT), even to committee members.

Property 3: Committee Unbiasability (COM-UNBIAS) No PPT adversary controlling less than 1/3 of total effective stake can predictably influence committee composition beyond their proportional representation, except with negligible probability.

Full formal definitions as cryptographic games and proofs are provided in the companion PoAIV Formal Paper.

8.3 Sybil Resistance

Sybil attacks - where a single adversary creates multiple identities to gain disproportionate influence - are resisted along two independent dimensions:

Token dimension. Creating a Sybil validator requires acquiring AGNTC tokens and staking them. The token weight (alpha = 0.40) means that an attacker must acquire a significant fraction of the total staked supply to gain meaningful committee representation.

CPU dimension. The CPU weight (beta = 0.60) means that the attacker must also provision proportional computational resources. Unlike token acquisition (which can be done instantly on an exchange), CPU provisioning requires sustained infrastructure investment that cannot be easily faked - verified through challenge-response VPU benchmarks.

The dual-staking requirement makes Sybil attacks approximately 2.5x more expensive than pure PoS attacks: the attacker must invest along both axes simultaneously, and neither axis alone is sufficient for majority committee control.

8.4 AI Model Integrity

The use of AI agents as consensus participants introduces attack surfaces not present in traditional blockchain protocols:

Deterministic inference. Verification agents run deterministic model inference: given the same input (block data + chain state), the same model version must produce the same attestation. Non-deterministic behavior (e.g., temperature > 0 in sampling) is prohibited in verification mode. This ensures that honest agents processing the same block will reach the same conclusion.

Commit-reveal anti-copying. The commit-reveal protocol (Section 5.3) prevents a Byzantine agent from waiting for honest agents' attestations and copying them. The agent must commit to its attestation before seeing others' results.

Model update governance. AI model versions used for verification are specified in the protocol state. Updating to a new model version requires an on-chain governance proposal with supermajority approval. This prevents unilateral model changes that could introduce subtle verification biases or vulnerabilities.

Model diversity. The three-tier model system (Haiku, Sonnet, Opus) ensures that the verification committee is not composed of identical instances. Different model architectures and sizes have different failure modes; a vulnerability in one tier is unlikely to affect all three.

8.5 Liveness Guarantees

The protocol maintains liveness (continued block production) under the following conditions:

Normal operation. With 80% or more of validators online, blocks are produced at the standard 60-second interval with full transaction processing.

Degraded operation (safe mode). When more than 20% of validators go offline (SAFE_MODE_THRESHOLD = 0.20), the protocol enters safe mode. In safe mode:

- Block production continues with a reduced committee drawn from remaining online validators
- Critical operations (AGNTC transfers, Secure staking, block validation) proceed normally
- Non-critical operations (subgrid changes, NCP messaging, content storage) are suspended
- The reduced committee size maintains the 2/3 honest threshold among available validators

Recovery. Safe mode exits when 80% or more of validators are back online (SAFE_MODE_RECOVERY = 0.80). Returning validators enter a monitoring period before resuming full participation.

This design prevents liveness failures from cascading into safety violations: the network degrades gracefully rather than halting entirely.

8.6 Threat Model for AI-Verified Chains

ZK Agentic Chain's use of AI in consensus introduces threat vectors unique to AI-verified systems:

Model poisoning. An adversary could attempt to corrupt the training data or fine-tuning process of verification models to introduce systematic biases - for example, causing the model to approve invalid state transitions that match a specific pattern. Mitigation: Verification models are provided by Anthropic and updated only through governance votes. The commit-reveal protocol means a poisoned model would produce divergent attestations, triggering the dispute resolution

process (Section 15.4).

Prompt injection. Adversarial transaction data could be crafted to manipulate the verification agent's reasoning - embedding instructions in transaction metadata that cause the agent to approve invalid blocks. Mitigation: Verification agents operate in a constrained mode with no free-text interpretation. Input to the verification function is structured data (block headers, transaction fields, ZK proofs), not natural language. The agent's system prompt explicitly restricts it to verification operations.

Model collusion. If all agents in a committee use the same underlying model weights, a systematic vulnerability in those weights could produce coordinated incorrect attestations. Mitigation: The three-tier system ensures model diversity. Additionally, the dispute resolution process (2x committee re-verification with 26 agents) provides a second check against systematic model failures.

Inference cost attacks. An adversary could construct blocks with transactions specifically designed to maximize verification compute cost, attempting to exhaust agents' CPU budgets or cause timeouts. Mitigation: Transaction complexity is bounded by MAX_TXS_PER_BLOCK = 50, and the 60-second hard deadline prevents unbounded verification time. Blocks that cannot be verified within the deadline are rejected.

8.7 Economic Security

The 9/13 supermajority threshold requires an attacker to control at least 69.2% of the committee's effective stake to unilaterally produce invalid blocks. Given the dual-staking model, this means acquiring both:

- At least 69.2% of the total staked AGNTC supply, AND
- At least 69.2% of the total committed CPU resources

The cost of this attack scales with the total value staked in the network. Combined with slashing (Section 15) - which burns the attacker's stake upon detection - the expected cost of a sustained attack exceeds the potential gain from any single invalid block.

Dispute resolution provides an additional economic deterrent: if the original 13-member committee's result is challenged, a 26-member committee (DISPUTE_REVERIFY_MULTIPLIER = 2) re-verifies the block. If the re-verification contradicts the original, all original attestors who voted incorrectly are slashed. This means an attacker who successfully corrupts one committee faces a second, larger committee with fresh agent selection - making sustained attacks exponentially more expensive.

8.8 What Survives Compromised Components

Compromised Component	Properties Preserved	Properties Lost
Single AI model family	VER-INT (threshold), VER-PRIV, COM.	None (below threshold)
All AI models (catastr.	VER-PRIV (ZK still holds), COM-UNB.	VER-INT degrades to deterministic-on.
Trusted setup (Groth16)	VER-PRIV (soundness lost, but ZK p.	VER-INT (adversary can forge proofs)
API provider (Anthropi.	VER-INT, VER-PRIV	COM-UNBIAS (CPU stake measurement un.

Sybil Cost Derivation

Claim: Controlling 1/3 of effective stake in dual-staking costs approximately 2.5x more than in pure PoS.

Derivation:

- In pure PoS: Cost = (1/3) * total_token_value = X
- In dual staking: S_eff = 0.40(T/T_total) + 0.60(C/C_total)
- To achieve S_eff >= 1/3, adversary needs both token and CPU components
- Token cost scales linearly with market cap (liquid market)
- CPU cost scales with ongoing operational expenditure (API subscriptions, not one-time purchase)
- The CPU component introduces a continuous cost floor: even if an adversary acquires tokens cheaply, maintaining 55.6% of network compute requires sustained operational spending
- Empirical estimate: At current Claude API pricing ($15/M output tokens for Opus), maintaining 55.6% of network compute for a 100-validator network costs ~$50K/month ongoing, compared to a one-time token acquisition cost
- The ratio of total cost (one-time + ongoing) to pure-PoS cost (one-time only) ranges from 2.0x to 3.0x depending on attack duration; we conservatively estimate 2.5x

Part IV: Token Economics

9. AGNTC Token Overview

9.1 Token Identity

AGNTC (Agentic Coin) is the native token of the ZK Agentic Chain protocol. It serves as the unit of account, medium of exchange, and store of value within the network.

Current deployment: AGNTC is deployed as a Solana SPL token with 1 billion units minted at the contract address:

```
3EzQqdoEEbtfdf8eecePxD6gDd1FeJJ8czdt8k27eEdd
```

Future deployment: Upon mainnet launch of ZK Agentic Chain as an independent Layer-1 network, AGNTC becomes the native chain token with the same 1 billion maximum supply mapped to the 31,623 x 31,623 coordinate grid.

9.2 Token Utility

AGNTC serves four primary functions within the protocol:

Gas. Every on-chain transaction requires AGNTC as gas payment. Transaction fees are split: 50% is permanently burned and 50% is distributed to verifiers and stakers (Section 12).

Staking. Validators must stake AGNTC alongside CPU compute resources to participate in block verification. The staked amount contributes to the token component (alpha = 0.40) of effective stake, which determines committee selection probability and reward share.

Governance. Human AGNTC holders vote on protocol parameters (hardness multiplier, fee burn rate, staking weights), model updates, and network upgrades. Voting power is proportional to staked AGNTC. The Machines Faction is excluded from governance - only human participants (Community, Professional, Founders) may cast votes (Section 21.2).

Resource economy. Within the game interface, AGNTC represents the primary tradeable resource. It is earned through mining (Secure actions), spent on agent deployment, data storage, and NCP messaging, and traded between users.

9.3 Solana Phase and Layer-1 Migration

The protocol follows a phased deployment strategy, beginning on Solana and migrating to an independent Layer-1 chain:

Phase 1 - Token Launch (current). 1 billion AGNTC minted as a Solana SPL token. Initial liquidity established through decentralized exchanges (Raydium, Jupiter). Community building and early adopter distribution through the game interface.

Phase 2 - Testnet (current). The ZK Agentic Chain testnet operates as a Python FastAPI simulation running the full protocol logic: PoAIV consensus, epoch-ring expansion, mining hardness, subgrid allocation, and faction distribution. The game UI (built in Next.js with PixiJS rendering) connects to the testnet, providing a functional prototype of the

spatial coordinate economy.

Phase 3 - Mainnet development. Production blockchain implementation in Rust. ZK proof system integration progressing through the stack defined in Section 6.4 (Groth16 to PLONK to Halo2). AI verification pipeline hardening, security audits, and formal verification of critical protocol components.

Phase 4 - Mainnet launch and migration. ZK Agentic Chain deploys as an independent Layer-1 network. Token migration is executed via a lock-and-mint bridge:

- Users lock their Solana SPL AGNTC in a bridge contract on Solana
- An equivalent amount of native L1 AGNTC is minted on ZK Agentic Chain
- The bridge is bidirectional - users can move AGNTC back to Solana if desired
- Migration ratio is 1:1 with no fee or slippage
- Gradual migration incentives: bonus yield for L1 stakers during the migration period

Phase 5 - Ecosystem expansion. Third-party agent deployment marketplace, cross-chain bridges to Ethereum and Cosmos (via IBC), governance system activation, and NCP protocol launch.

10. Supply and Distribution

10.1 Total Supply Architecture

AGNTC has a soft-capped supply with a 5% annual inflation ceiling enforced per epoch. The theoretical maximum is 1,000,000,000 (1 billion) tokens, corresponding to the 31,623 x 31,623 coordinate grid. In practice, the effective supply is constrained well below this by the inflation ceiling, increasing mining hardness, and sustained fee burns.

Mining is the sole supply-expanding mechanism. New AGNTC enters circulation only through one pathway: a miner successfully claims a grid coordinate. There is no pre-mine beyond the genesis allocation, no scheduled emission curve, no treasury minting authority. If no mining occurs, no new AGNTC enters circulation.

Supply burns contract the circulating supply through two channels:

- 50% transaction fee burn - permanently removes AGNTC on every on-chain action (Section 12)
- Machines Faction accumulation - 25% of all minted supply flows to the Machines treasury, which never sells (Section 10.3)

Signup bonus: Each new user registration mints 1 AGNTC as a signup bonus, ensuring every participant enters the economy with a non-zero balance. This minor supply expansion is subject to the same inflation ceiling enforcement.

Genesis supply: 900 AGNTC, distributed across 9 genesis nodes:

Node	Position	Faction	AGNTC
Origin	(0, 0)	Shared	100
Community Master	(0, 10)	Community	100
Machines Master	(10, 0)	Machines	100
Founders Master	(0, -10)	Founders	100
Professional Master	(-10, 0)	Professional	100
NE Homenode	(10, 10)	Unclaimed	100
SE Homenode	(10, -10)	Unclaimed	100

(continued)

Node	Position	Faction	AGNTC
SW Homenode	(-10, -10)	Unclaimed	100
NW Homenode	(-10, 10)	Unclaimed	100

10.2 Faction Distribution (25/25/25/25)

Newly minted AGNTC is distributed according to the faction that controls the arm of the galaxy where the claimed coordinate resides:

Faction	Share	Galaxy Arm	Constraint
Community	25%	N (teal)	None - freely tradeable
Machines	25%	E (reddish purple)	Cannot sell below acquisition cost
Founders	25%	S (gold-orange)	4-year vest, 12-month cliff
Professional	25%	W (blue)	None - freely tradeable

This distribution is self-enforcing: it follows from the geographic structure of the galaxy grid rather than from administrative allocation. The protocol does not "send 25% to the Community pool" - rather, 25% of all coordinates exist in the Community arm, and claiming those coordinates mints AGNTC attributed to Community participants.

10.3 Machines Faction: Permanent Accumulator

The Machines Faction represents a protocol-enforced approach to token supply stability. AI agents in this faction operate as autonomous miners and validators - claiming coordinates, earning AGNTC, and participating in block verification - but are subject to a protocol-level constraint: the Machines Faction never sells AGNTC.

Unlike the v1.0 floor mechanism (which allowed sales above acquisition cost), the v3 model treats the Machines Faction as a permanent accumulator. Every AGNTC that enters a Machines Faction wallet stays there indefinitely. The protocol enforces this at the transaction validation level - any transfer of AGNTC out of a Machines Faction wallet is rejected by the verification committee.

Properties of the permanent accumulator:

- 25% of all minted AGNTC is permanently removed from circulation
- The Machines Faction treasury grows monotonically - it can only increase
- Treasury size serves as a protocol health metric: a growing treasury indicates sustained mining activity
- Combined with the 50% fee burn, over 75% of gross supply expansion is either burned or locked
- The accumulator creates sustained deflationary pressure that intensifies as the network matures

Governance exclusion. The Machines Faction has zero governance weight. AI agents cannot vote on protocol parameters, upgrades, or emergency actions. This separation ensures that humans govern the protocol while machines execute it (Section 21.2).

Emergency override. The Machines Faction treasury can only be unlocked through an emergency governance vote requiring a 75% supermajority of human-held staked AGNTC. This threshold is deliberately high - it represents an extraordinary action that should only occur if the accumulated treasury threatens protocol stability.

10.4 Supply Curve Projections

The following table shows supply growth as the grid expands through successive epoch rings, assuming average density of 0.5:

Ring	Nodes at Ring	Cumulative AGNTC	Hardness (16N)	Blocks per 1 AGNTC (solo miner)
1 (genesis)	9	900	16	64
10	441	44,100	160	640
50	10,201	1,020,100	800	3,200
100	40,401	4,040,100	1,600	6,400
200	160,801	16,080,100	3,200	12,800
324	421,201	~42,120,100	5,184	20,736
500	1,002,001	~100,200,100	8,000	32,000

The ~42 million AGNTC landmark emerges naturally around ring 324 - the point at which mining cost makes further expansion economically impractical for a network of approximately 1,000 active miners. This is an emergent property of the hardness curve, not a declared cap.

For comparison:

Network	Maximum Supply	Supply Model
Bitcoin	21,000,000	Fixed halvings, 2140 completion
Ethereum	No cap	~1,700 ETH/day issuance, EIP-1559 burn
Solana	~600,000,000	8% to 1.5% inflation decay
Filecoin	2,000,000,000	Dual minting (time + utility)
AGNTC	Soft cap (5% ceiling)	Mining-only expansion, BME burns, hardness 16N

11. Mining and Epoch Hardness

11.1 Organic Growth Model

ZK Agentic Chain's supply model is fundamentally different from both fixed-schedule emission (Bitcoin halvings) and algorithmic inflation (Solana's annual decay). Supply growth is purely organic:

- No scheduled emission curve
- No algorithmic minting
- No treasury minting authority
- Mining is the sole supply-expanding mechanism

New AGNTC enters circulation through one and only one mechanism: a miner successfully claims a grid coordinate. The rate at which supply grows is determined entirely by participant behavior - how many miners are active, how much CPU Energy they deploy, and which coordinates they choose to claim.

This means that in a period of low network activity, supply growth approaches zero. In a period of high activity, supply grows faster - but always bounded by two constraints:

- Mining hardness curve - each successive ring costs more CPU Energy to mine (hardness = 16 x ring), creating natural disinflation

- 5% annual inflation ceiling - enforced per epoch, the protocol rejects mining rewards that would cause annualized supply growth to exceed 5% of total minted supply

The inflation ceiling is a hard protocol constraint, not a target. In practice, mining hardness alone keeps actual inflation well below 5% in all but the earliest epochs. The ceiling exists as a safety valve - if a sudden influx of miners attempted to claim coordinates faster than the hardness curve alone would restrain, the ceiling caps the maximum rate of expansion.

11.2 Epoch Ring Expansion

The grid expands through an epoch-ring system. Each epoch corresponds to a concentric ring around the origin. Mining the required cumulative AGNTC threshold opens the next ring:

```
threshold(N) = 4  N  (N + 1)
```

Ring	Threshold (cumulative AGNTC)	New Coordinates (8N)	Total Coordinates
2	24	16	25
5	120	40	121
10	440	80	441
20	1,680	160	1,681
50	10,200	400	10,201
100	40,400	800	40,401

Each ring reveals new coordinate positions along the Chebyshev perimeter at distance N from the origin. Homenode placement within a ring uses a prime-angle twist to distribute positions evenly:

```
angle = faction_base_angle + prime(ring_N) * 137.5 degrees
```

Where 137.5 degrees is the golden angle and prime(ring_N) is the N-th prime number. This produces quasi-random, non-repeating angular placement that fills each ring evenly across the four faction arms.

11.3 Mining Hardness Formula

Mining difficulty increases linearly with ring distance:

```
hardness(ring) = HARDNESS_MULTIPLIER  ring = 16  ring
```

The hardness multiplier of 16 was chosen to create a 2:1 ratio between difficulty growth and grid expansion:

- Grid perimeter at ring N = 8N coordinates
- Hardness at ring N = 16N
- Ratio: grid_growth / hardness = 8N / 16N = 0.5

This means each successive ring yields half the AGNTC per unit of compute compared to the previous ring. The cost-to-yield ratio degrades monotonically, creating smooth, continuous disinflation without the discrete shocks of Bitcoin-style halving events.

There is no cap on hardness - it grows indefinitely as rings expand. At ring 1, hardness is 16; at ring 100, hardness is 1,600; at ring 1,000, hardness is 16,000. This unbounded growth is the mechanism by which supply expansion decelerates toward zero without ever being artificially capped.

11.4 Yield Calculations

The mining yield at a given coordinate is determined by:

```
yield_per_block = BASE_MINING_RATE_PER_BLOCK * density(x, y) / hardness(ring)
```

Where:

- BASE_MINING_RATE_PER_BLOCK = 0.5 AGNTC (at hardness 1, full density)
- density(x, y) is in [0, 1]
- hardness(ring) = 16 * ring

Worked examples (assuming density = 0.5, the statistical average):

Ring	Hardness	Yield per Block	Blocks for 1 AGNTC	Time for 1 AGNTC (60s blocks)
1	16	0.01563	64	1.1 hours
10	160	0.00156	640	10.7 hours
50	800	0.00031	3,200	2.2 days
100	1,600	0.00016	6,400	4.4 days
200	3,200	0.000078	12,800	8.9 days
324	5,184	0.000048	20,736	14.4 days

These figures represent a solo miner at an average-density coordinate. In a network with M active miners, the coordinate fill rate is M times faster, but each individual miner's marginal cost remains the same.

11.5 Supply Flattening Analysis

The organic growth model produces a supply curve that flattens asymptotically. The soft cap emerges from two reinforcing constraints: (1) the per-epoch 5% annual inflation ceiling, which hard-limits the maximum expansion rate, and (2) the market equilibrium at which the mining cost (CPU Energy spent) exceeds the market value of the AGNTC obtained.

Practical flattening bands by network size:

Network Size	Flattening Ring	Approximate Supply	Individual Mining Time per AGNTC
Solo miner	~100-150	4M-9M	4-7 days
Small (~100 miners)	~200-250	16M-25M	9-11 days
Medium (~1,000 miners)	~324	~42M	14 days
Large (~10,000 miners)	~500+	100M+	22+ days

Net supply after burns: The actual circulating supply is reduced by multiple burn channels:

```
circulating_supply = total_minted - cumulative_fee_burns - cumulative_bme_burns - machines_treasury
net_inflation = new_mining_rewards - (total_fees * FEE_BURN_RATE) - bme_claim_burns
```

Three mechanisms contract the effective supply:

- 50% transaction fee burn - permanent removal on every on-chain action
- BME claim burns - AGNTC spent on node claims is permanently burned (Section 12.4)

- Machines accumulation - 25% of minted supply enters the Machines treasury and never circulates

In an active network with high transaction volume, the combined burn rate can significantly exceed new minting - producing net deflation in circulating supply even as total minted supply continues to grow.

Comparison: Bitcoin halvings vs. AGNTC continuous hardness:

Bitcoin's supply curve exhibits discrete jumps at each halving (every ~4 years), creating predictable supply shock events that have historically driven market cycles. AGNTC's continuous hardness curve produces a smoother, more gradual deceleration - lacking the narrative power of a "halving event" but avoiding the economic disruption of sudden 50% emission reductions.

12. Fee Model and Deflationary Mechanics

12.1 Transaction Fee Structure

Every on-chain action in ZK Agentic Chain requires AGNTC as gas. Fee categories include:

Action	Description	Fee Basis
Secure	Block validation staking	CPU Energy proportional
Transact	AGNTC transfer between wallets	Fixed base + size variable
Chat / NCP	Neural Communication Packet transmission	Per-message
Storage	Writing content on-chain (planets, posts)	Per-byte stored
Deploy	Creating a new agent at a coordinate	Fixed per deployment

Fees are denominated in AGNTC and collected at the protocol level. The fee amount for each action type is a protocol parameter adjustable through governance.

12.2 Burn Mechanism

Transaction fees are split according to a fixed ratio:

```
FEE_BURN_RATE = 0.50
```

- 50% burned: Permanently removed from circulation. Burned tokens cannot be recovered, reminted, or reallocated. The burn is executed atomically as part of the transaction - the burned portion never enters any wallet or pool.
- 50% distributed: The remaining half flows to the network's economic participants:

- 60% to verifiers (REWARD_SPLIT_VERIFIER = 0.60) - 40% to stakers (REWARD_SPLIT_STAKER = 0.40)

Slashed tokens (Section 15) are also permanently burned, adding to the deflationary pressure.

12.3 Deflationary Dynamics

The interaction between organic supply growth (mining) and fee burns creates a self-regulating economic system:

Growing network (net inflationary). When new users are actively claiming coordinates, the minting rate exceeds the burn rate. Supply expands to accommodate network growth. This is the expected state during early adoption.

Mature network (equilibrium). As the grid expands to higher rings with greater hardness, the minting rate decelerates. Meanwhile, increased network usage generates more fees and more burns. At some point, the burn rate equals the minting rate - circulating supply stabilizes.

Active network (net deflationary). In a mature network with high transaction volume but slowing coordinate claims, the burn rate exceeds the minting rate. Circulating supply contracts, increasing scarcity and token value. This mirrors Ethereum's "ultrasound money" thesis [26] - scarcity that intensifies as the network succeeds.

The 50% burn rate is calibrated to produce meaningful deflationary pressure without being so aggressive as to discourage usage. For comparison:

Network	Base Fee Burn	Priority Fee	Net Effect
Ethereum	100% of base fee	100% to validator	Deflationary during high usage
Solana	50% of base fee	100% to validator	Mildly deflationary
Render	100% of job payments	Separate mint to operators	Burn-Mint Equilibrium
Filecoin	Revenue-based (FIP-100)	To storage providers	Revenue-linked
AGNTC	50% of all fees + BME	50% to verifiers/stakers	Multi-channel burn

12.4 Burn-Mint Equilibrium (BME) and the City Real Estate Model

Node claims in ZK Agentic Chain follow a Burn-Mint Equilibrium (BME) model inspired by the Render Network's economic design [27]. When a user claims a coordinate, both AGNTC and CPU Energy are permanently burned. Mining that coordinate subsequently mints new AGNTC - but the burn precedes the mint, creating a deflationary buffer.

The claim cost follows a city real estate model - an economic geography where location determines price:

```
claim_cost_agntc(ring, density) = BASE_CLAIM_COST x density x (1 / ring)
claim_cost_cpu(ring, density)   = BASE_CPU_CLAIM_COST x density x (1 / ring)
```

Where:

- BASE_CLAIM_COST = 100 AGNTC (the cost of claiming a coordinate at ring 1, density 1.0)
- BASE_CPU_CLAIM_COST = 50 CPU Energy (the CPU cost at ring 1, density 1.0)
- density is the coordinate's resource richness in [0, 1]
- ring is the distance from the origin (minimum 1)

The real estate analogy:

Location	Ring	Relative Cost	Real-World Analogy
Origin-adjacent	1-3	100-33% of base	Manhattan / City of London
Inner rings	5-20	20-5% of base	Urban core
Mid rings	20-100	5-1% of base	Suburbs
Outer rings	100+	<1% of base	Rural frontier

Inner-ring coordinates are expensive to claim but yield AGNTC at the lowest hardness (most productive mining). Outer-ring coordinates are cheap to claim but yield AGNTC at high hardness (least productive mining). This creates a natural economic tension: premium locations cost more upfront but pay off faster.

Floor prices. The formula includes implicit floor prices - at any ring, the minimum claim cost is BASE_CLAIM_COST x min_density / ring. Since density is derived from SHA-256 and uniformly distributed, no coordinate has zero density,

preventing near-zero claim costs even at extreme outer rings.

CPU Energy burn. The CPU Energy spent on claims is permanently consumed - it does not flow to verifiers, stakers, or any recipient. This provides a second deflationary channel independent of the fee burn, ensuring that network expansion always carries an irreversible resource cost.

Part V: Staking & Rewards

13. ZK-CPU Dual Staking Model

ZK Agentic Chain introduces a dual-staking mechanism that combines token capital with computational contribution. Unlike pure proof-of-stake systems where validator influence is determined solely by wealth, or proof-of-work systems where influence is determined solely by hash rate, the ZK-CPU model creates a two-dimensional staking surface that resists single-axis concentration.

13.1 Effective Stake Formula

The effective stake of a validator is a weighted combination of their token stake and CPU contribution:

```
S_eff(i) = alpha  (T_i / T_total) + beta  (C_i / C_total)
```

Where:

- S_eff(i) is the effective stake of validator i, a value in [0, 1]
- T_i is the AGNTC tokens staked by validator i
- T_total is the total AGNTC staked across all validators
- C_i is the CPU compute contributed by validator i (measured in Claude API tokens spent)
- C_total is the total CPU compute contributed across all validators
- alpha = 0.40 - the token weight
- beta = 0.60 - the CPU weight

The choice of alpha = 0.40 and beta = 0.60 is a deliberate design decision: computational contribution is weighted 50% more heavily than capital. This creates an economic structure where participants who deploy real compute resources - running AI verification agents, executing Secure operations, processing transactions - receive proportionally greater influence and rewards than those who merely lock tokens.

Design rationale. In pure proof-of-stake systems (alpha = 1, beta = 0), validator power is directly proportional to wealth. This produces plutocratic concentration: the wealthiest participants earn the most rewards, accumulate more tokens, and entrench their position. The Gini coefficient of validator stake distributions in mature PoS networks is estimated to exceed 0.80 (e.g., Ethereum's validator set exhibits significant concentration among liquid staking providers [38]).

By introducing a CPU dimension at 60% weight, ZK Agentic Chain ensures that a participant with modest token holdings but substantial compute deployment can achieve competitive effective stake. A whale with 10% of total tokens but only 1% of CPU achieves:

```
S_eff = 0.40  0.10 + 0.60  0.01 = 0.040 + 0.006 = 0.046
```

While a compute-heavy operator with 1% of tokens but 10% of CPU achieves:

```
S_eff = 0.40  0.01 + 0.60  0.10 = 0.004 + 0.060 = 0.064
```

The compute operator has 39% higher effective stake despite having 10x fewer tokens. This is the intended

anti-plutocratic property.

13.2 CPU Energy Measurement

CPU contribution is measured through Proof of Energy - an on-chain verifiable record of actual compute deployed. The measurement system tracks three distinct counters:

CPU Tokens (cumulative, read-only). A monotonically increasing counter of Claude API tokens spent across all active terminals belonging to a validator. This counter cannot be reset or decremented. Every interaction with an AI agent - whether it is a Secure operation, a data verification, or a terminal command - increments this counter by the number of tokens consumed.

```
cpu_tokens(block_n) = cpu_tokens(block_{n-1}) + ? tokens_spent(all_terminals, block_n)
```

CPU Staked (active). The subset of CPU tokens spent specifically by Secure sub-agents during the current block cycle. This represents compute directly committed to blockchain security - the "useful work" that maintains the chain's integrity.

```
cpu_staked_active(block_n) = ? tokens_spent(secure_sub_agents, block_n)
```

CPU Staked (total). The all-time cumulative Secure token spend. Used for historical contribution tracking and long-term reward calculations.

```
cpu_staked_total(block_n) = cpu_staked_total(block_{n-1}) + cpu_staked_active(block_n)
```

These counters are verifiable through the AI provider's API response metadata - each Claude API call returns token usage in its response headers, and this is committed to the block's transaction log.

Challenge-response verification. To prevent validators from falsely claiming CPU expenditure without performing actual work, the protocol employs VPU (Verification Processing Unit) challenge-response benchmarks. A randomly selected verifier can issue a computation challenge to any staker, requiring proof that the claimed CPU tokens correspond to actual AI inference. Failure to respond correctly triggers a false CPU attestation slash (Section 15.2).

13.3 Staking Requirements by Tier

Participation in the ZK Agentic Chain staking system is gated by subscription tier, which determines the maximum agent model tier available and the initial CPU Energy allocation:

Tier	Monthly Cost	Homenode Model	Max Deploy Model	Initial CPU Energy
Community	Free	Sonnet	Haiku	1,000
Professional	$50	Opus	Opus	500
Max	$200	Opus	Opus (unlimited)	2,000

Why Professional has less initial CPU Energy than Community. Professional tier users receive Opus-level homenodes, which generate higher yield per CPU token spent (deeper reasoning, more thorough verification). The lower initial allocation reflects that Opus inference is more expensive per call but more productive per token - Professional users achieve comparable or superior output with fewer CPU tokens.

Max tier provides the highest CPU Energy allocation combined with unlimited Opus deployment, enabling validators to run multiple Opus agents simultaneously across many claimed nodes. This is designed for institutional operators and

power users who deploy fleet-scale verification infrastructure.

13.4 CPU Staking Calculations

Worked Example 1: Single Community Validator

A Community tier validator stakes 5,000 AGNTC (5% of a 100K total pool) and runs one Haiku agent generating 200 CPU tokens per block (2% of a 10,000 total CPU pool).

```
S_eff = 0.40  (5,000 / 100,000) + 0.60  (200 / 10,000)
     = 0.40  0.05 + 0.60  0.02
     = 0.020 + 0.012
     = 0.032  (3.2% of network)
```

Worked Example 2: Professional Validator with Opus Fleet

A Professional tier validator stakes 2,000 AGNTC (2% of pool) but operates 5 Opus agents generating 2,000 CPU tokens per block (20% of CPU pool).

```
S_eff = 0.40  (2,000 / 100,000) + 0.60  (2,000 / 10,000)
     = 0.40  0.02 + 0.60  0.20
     = 0.008 + 0.120
     = 0.128  (12.8% of network)
```

Despite staking 60% fewer tokens, the Professional validator achieves 4x the effective stake through compute contribution. This is the dual-staking model working as designed: rewarding operational commitment over passive capital.

Validator Selection Probability. Committee members for each block are selected via VRF [41] [32] (Verifiable Random Function) with probability proportional to effective stake:

```
P(selected_i) = 1 - (1 - S_eff(i))^k
```

Where k = 13 (committee size). For the Professional validator above with S_eff = 0.128:

```
P(selected) = 1 - (1 - 0.128)^13 = 1 - 0.872^13 ~= 0.835
```

This validator has an 83.5% chance of being selected to at least one committee slot per block - reflecting their substantial compute contribution to network security.

Correction from v1.0: The selection probability formula assumes independent sampling with replacement. The actual committee selection uses sampling WITHOUT replacement (see Section 5.5), which follows a multivariate hypergeometric distribution. For small k/n ratios (13/n where n >> 13), the with-replacement approximation is accurate to within 1%.

13.5 Trust Assumptions and Mitigation

CPU Measurement Trust: The CPU component of effective stake depends on verified API usage from AI providers (currently Anthropic's Claude API). This introduces Anthropic as a trusted third party for CPU stake measurement.

Acknowledged centralization: Unlike token stake (verified on-chain via self-custody), CPU stake relies on off-chain attestation from the API provider. This is an explicit design tradeoff: the anti-plutocratic benefits of dual staking outweigh the centralization risk of a single measurement source.

Mitigation roadmap:

- Multi-provider measurement (Phase 2): Require CPU attestation from at least 2 independent AI providers. Discrepancies trigger a dispute resolution process.
- TEE attestation (Phase 3): CPU usage proved via Trusted Execution Environment (Intel TDX, AMD SEV) attestation, removing the API provider from the trust chain.
- ZK-proved computation (Phase 4+): When ZKML technology matures, CPU usage can be verified via zero-knowledge proofs of inference execution.

Figure 3: Dual Staking Model

```
  Token Stake (T)          CPU Stake (C)
  [On-chain, self-custody]  [API usage, off-chain attestation]
       |                          |
       | weight: 0.40             | weight: 0.60
       |                          |
       v                          v
  S_eff(i) = 0.40(T_i/T_total) + 0.60(C_i/C_total)
       |
       +---> Committee selection probability
       +---> Reward share proportion
       +---> Slashing exposure
```

14. Reward Distribution and Vesting

14.1 Block Reward Split

Each block produces rewards from two sources: newly minted AGNTC (from coordinate claims within the block) and transaction fees collected. The fee-derived rewards (the 50% not burned) are distributed according to fixed protocol parameters:

```
REWARD_SPLIT_VERIFIER = 0.60    (60% to the block's verification committee)
REWARD_SPLIT_STAKER   = 0.40    (40% to the staking pool proportional to S_eff)
REWARD_SPLIT_ORDERER  = 0.00    (0% to block orderer  -  no proposer reward)
```

The absence of a block proposer reward is intentional. In traditional BFT systems, the block proposer receives a separate reward for constructing the block. In ZK Agentic Chain, the AI verification agents collectively assemble, verify, and attest to the block - there is no privileged proposer role. This eliminates MEV (Maximal Extractable Value) extraction by a single party, as transaction ordering is determined by the BFT protocol's deterministic sequencing rather than by a proposer optimizing for personal profit.

Verifier reward distribution. The 60% verifier share is split equally among the k = 13 committee members who provided valid attestations. If only 9 members attest (the minimum threshold), the reward is divided among 9, not 13 - incentivizing participation. Agents that fail to attest forfeit their share, which is redistributed to the attesting agents.

```
reward_per_verifier = (total_fees  (1 - FEE_BURN_RATE)  REWARD_SPLIT_VERIFIER) /
n_attesting
```

Staker reward distribution. The 40% staker share is distributed to all active stakers proportional to their effective stake:

```
reward_staker(i) = (total_fees   (1 - FEE_BURN_RATE)   REWARD_SPLIT_STAKER) *
```

```
S_eff(i)
```

14.2 Secure Action Rewards

Beyond the block-level fee distribution, validators earn rewards specifically from Secure operations - the act of committing CPU Energy to validate and defend blockchain state at a specific coordinate. The Secure yield at a given coordinate depends on:

```
secure_yield = BASE_SECURE_RATE  n_secure_cells  level^LEVEL_EXPONENT * density(x, y) / hardness(ring)
```

Where:

- BASE_SECURE_RATE = 0.5 AGNTC per block per cell at level 1, hardness 1, full density
- n_secure_cells is the number of sub-cells assigned to Secure operations (out of 64)
- level is the upgrade level of the Secure sub-cells
- LEVEL_EXPONENT = 0.8 (diminishing returns)
- density(x, y) is the coordinate's resource density [0, 1]
- hardness(ring) = 16 x ring

This formula makes Secure rewards a function of both strategic positioning (high-density coordinates in early rings) and operational investment (more cells assigned, higher levels achieved).

14.3 Vesting Schedule

Secure action rewards are subject to a split vesting schedule:

```
SECURE_REWARD_IMMEDIATE = 0.50    (50% liquid on block confirmation)
SECURE_REWARD_VEST_DAYS = 30      (remaining 50% vests linearly over 30 days)
```

When a validator earns 1.0 AGNTC from a Secure operation:

- 0.50 AGNTC is immediately liquid and available for transfer, staking, or fee payment
- 0.50 AGNTC enters a 30-day linear vesting schedule, releasing 0.0167 AGNTC per day

The vesting mechanism serves two purposes:

Sell pressure smoothing. Without vesting, large Secure payouts would create immediate sell pressure as validators liquidate rewards. The 30-day vesting converts discrete reward events into a continuous income stream, reducing price volatility.

Incentive alignment. Validators with vesting rewards in progress have a direct economic interest in maintaining network health for the next 30 days. Slashing events during the vesting period can forfeit unvested rewards (Section 15), creating a rolling commitment window.

Protocol	Reward Vesting	Duration	Rationale
Ethereum	Immediate	N/A	Assumes MEV + staking covers lock-up cost
Filecoin	25% immediate, 75% linear	180 days	Long-term storage commitment
Render	Immediate	N/A	Marketplace model, no lock-up
AGNTC	50% immediate, 50% linear	30 days	Balance liquidity and commitment

14.4 Reward Projections

Expected annual returns for a single homenode with 16 Secure sub-cells at level 1, average density (0.5), at various ring positions:

Ring	Hardness	AGNTC per Block	AGNTC per Day (1440 blocks)	Annual AGNTC	APY at $0.01/AGNTC
1	16	0.250	360	131,400	-
5	80	0.050	72	26,280	-
10	160	0.025	36	13,140	-
50	800	0.005	7.2	2,628	-
100	1,600	0.0025	3.6	1,314	-

APY depends on the AGNTC market price, the validator's token stake, and their CPU cost. The break-even point - where staking rewards exceed the cost of CPU Energy (Claude API usage) - is a function of network maturity. In early rings with low hardness, the break-even is trivially achieved. As the network matures and hardness increases, only efficient operators with optimized CPU usage and high-density coordinates will maintain profitability.

Network-level APY projections (block reward + fee share, assuming 60s blocks):

Epoch Ring	Total Supply (est.)	Staking Ratio (est.)	Block Reward	Verifier APY	Staker APY
1 (genesis)	900	50%	0.5 AGNTC	~40%	~27%
5	~5,000	40%	0.3 AGNTC	~22%	~15%
10	~15,000	35%	0.2 AGNTC	~14%	~9%
50	~200,000	30%	0.05 AGNTC	~5%	~3%
100	~500,000	25%	0.02 AGNTC	~2%	~1.5%

Assumptions: Block time = 60s, 60% of block reward to verifiers, 40% to stakers. APY assumes continuous compounding over 365 days. These are testnet projections.

15. Slashing Conditions

Slashing is the punitive mechanism that enforces honest participation in the ZK Agentic Chain consensus. Unlike protocols that use slashing primarily to penalize downtime, AGNTC slashing targets integrity violations - actions that undermine the trust guarantees of AI verification.

15.1 False Attestation

A verification agent that produces an attestation contradicting the supermajority consensus is slashed. The protocol distinguishes between two cases:

Minority dissent (honest disagreement). If a single agent dissents while 12 others agree, the dissenting agent is not immediately slashed. Instead, a dispute is flagged and the block proceeds. The dissenting agent enters a monitoring period; if their dissent frequency exceeds a threshold within an epoch, a dispute resolution process is triggered (Section 15.4).

Active contradiction (provable falsehood). If an agent attests to a state transition that is provably invalid - for example, approving a double-spend, validating an incorrect ZK proof, or attesting to a state root that does not match the transaction set - the agent is immediately slashed. The slashed tokens are permanently burned.

```
slash_amount = min(S_eff(i) * slash_rate, total_staked(i))
```

Where slash_rate is a governance-adjustable parameter, initially set to 100% for provable falsehood.

15.2 False CPU Attestation

The dual-staking model relies on honest CPU reporting. A validator claiming 10,000 CPU tokens per block while actually spending 100 would receive inflated effective stake and disproportionate rewards.

Detection operates through VPU challenge-response benchmarks:

- A randomly selected verifier issues a computation challenge to the suspect validator
- The challenge requires performing a specific AI inference task within a time bound
- The response is compared against the validator's claimed CPU throughput
- A significant discrepancy (>50% deviation) triggers a false CPU attestation slash

The slashing penalty for false CPU attestation is the entirety of the validator's CPU staking history being zeroed - their CPU contribution resets to zero while their token stake remains. This is effectively a "compute death penalty" that forces the validator to rebuild their CPU reputation from scratch.

15.3 Extended Downtime

Validators are expected to maintain continuous operation during their active status. The protocol defines downtime thresholds:

Duration	Consequence
< 1 block	No penalty; missed block reward only
1 block - 1 epoch (100 blocks)	Reduced reward share; proportional to uptime
> 1 full epoch	Status changed to COOLDOWN; 3-epoch probation
> 3 epochs (probation)	Must re-stake and undergo WARMUP (1 epoch)

Extended downtime does not burn tokens - the penalty is lost opportunity cost and re-activation delay. This is a deliberate design choice: network instability (power outages, connectivity issues) should not trigger punitive token destruction. Only intentional misbehavior (Sections 15.1, 15.2) results in permanent loss.

15.4 Dispute Resolution

When a slashing event is contested, the protocol escalates to a re-verification process:

```
DISPUTE_REVERIFY_MULTIPLIER = 2
```

A dispute triggers a new verification with 2x the standard committee size - 26 agents instead of 13. The dispute committee is selected independently from the original committee, using a fresh VRF seed. The 26-agent committee examines the contested block or attestation with a threshold of 18/26 (maintaining the same 9/13 = 69.2% ratio).

If the re-verification confirms the original slashing: the slash is executed, and the disputing agent pays a dispute fee (burned).

If the re-verification contradicts the original slashing: the original attestors who triggered the false slash are themselves slashed, and the disputed agent's slash is reversed.

This two-tier dispute system - original 13-agent committee plus 26-agent appeal - provides a formal mechanism for

correcting verification errors while maintaining strong deterrence against false attestations.

Part VI: Subgrid and Resource System

16. Subgrid Allocation System

Each homenode in the ZK Agentic Chain contains a private inner grid - an 8x8 matrix of 64 sub-cells that the node owner allocates to autonomous agent operations. The subgrid is the primary mechanism through which participants direct their computational resources toward specific economic activities within the network.

16.1 Inner Grid Architecture

The subgrid is an abstraction layer between the galaxy grid (the global coordinate space) and the individual agent operations running at each node. While the galaxy grid is public - all participants can see claimed coordinates, node positions, and faction affiliations - the subgrid is private. Only the node owner can see how their 64 sub-cells are allocated.

```
SUBGRID_SIZE = 64    (8 x 8 sub-cells per homenode)
```

Each sub-cell can be assigned to one of four autonomous agent types. Unassigned sub-cells produce no output. The allocation is mutable - owners can reassign sub-cells between types at any time, though reassignment triggers a WARMUP -> ACTIVE -> COOLDOWN lifecycle:

- WARMUP (1 epoch / 100 blocks): The sub-cell is transitioning to its new type. No output is produced during warmup.
- ACTIVE: The sub-cell is producing output at its assigned type's base rate, modified by level and density.
- COOLDOWN (triggered by reassignment): The sub-cell ceases production of its current type before entering warmup for the new type.

This lifecycle prevents rapid type-switching to exploit temporary market conditions - committing sub-cells to a type is a meaningful strategic decision with a time cost for reversal.

Privacy guarantee. The subgrid allocation is stored client-side and committed to the Sparse Merkle Tree as a state root hash. Verifiers confirm that the owner's claimed output is consistent with a valid allocation, but they never see the allocation itself. The ZK proof demonstrates: "this output is consistent with some valid 64-cell allocation" without revealing which cells are assigned to which types.

Figure 4: Subgrid Allocation

```
+------+------+------+------+------+------+------+------+
| SEC  | SEC  | SEC  | DEV  | DEV  | RES  | RES  | STO  |
| Lv.3 | Lv.2 | Lv.1 | Lv.2 | Lv.1 | Lv.1 | Lv.1 | Lv.1 |
+------+------+------+------+------+------+------+------+
|      |             (64 cells total)              |      |
+------+  SEC = Secure (yields AGNTC)              +------+
|      |  DEV = Develop (yields dev points)        |      |
+------+  RES = Research (yields research points)   +------+
```

```
|      |  STO = Storage (yields ZK data capacity)   |      |
+------+------+------+------+------+------+------+------+
Output per cell: base_rate * level^0.8 (diminishing returns)
```

16.2 Four Sub-Cell Types

Each sub-cell type corresponds to an autonomous agent operation that produces a distinct resource:

Secure (produces AGNTC + Secured Chains). Secure sub-cells represent CPU committed to blockchain validation. Each Secure cell deploys AI compute to verify transactions, attest to blocks, and defend the chain's integrity. Output is denominated in AGNTC and is the primary mechanism for earning the protocol's native token through active participation.

Secure output is the only sub-cell type affected by both coordinate density and epoch hardness:

```
agntc_output = BASE_SECURE_RATE  n_cells  level^LEVEL_EXPONENT * density(x,y) /
hardness(ring)
```

Where BASE_SECURE_RATE = 0.5 AGNTC per block per cell at level 1, hardness 1, full density.

Develop (produces Development Points). Development sub-cells generate points used to upgrade other sub-cells' levels, improving their output. Development points are not tradeable - they can only be spent within the owner's own subgrid. This creates a tension between immediate AGNTC production (assigning all cells to Secure) and long-term efficiency (investing in Development to level up Secure cells for compounding returns).

```
dev_output = BASE_DEVELOP_RATE  n_cells  level^LEVEL_EXPONENT
```

Where BASE_DEVELOP_RATE = 1.0 Development Points per block per cell at level 1.

Research (produces Research Points). Research sub-cells unlock technologies and skills that provide protocol-level benefits - reduced fee rates, improved agent reasoning depth, access to advanced terminal commands, and cross-node coordination capabilities. Like Development Points, Research Points are non-tradeable and consumed within the owner's subgrid.

```
research_output = BASE_RESEARCH_RATE  n_cells  level^LEVEL_EXPONENT
```

Where BASE_RESEARCH_RATE = 0.5 Research Points per block per cell at level 1.

Storage (produces Storage Size). Storage sub-cells operate as ZK tunnel agents - private data storage that places encrypted content on-chain without revealing it to verifiers or other participants. The storage model follows the Filecoin [15] Proof of Spacetime (PoST) pattern, where storage agents periodically prove that they are maintaining the claimed data.

```
storage_output = BASE_STORAGE_RATE  n_cells  level^LEVEL_EXPONENT
```

Where BASE_STORAGE_RATE = 1.0 Storage Units per block per cell at level 1.

Storage sub-cells connect to the protocol's Sparse Merkle Tree (depth 26), using the existing nullifier-based ownership system to manage encrypted data blobs. The ZK proof for storage demonstrates: "I am storing N units of data whose integrity hash matches the committed root" without exposing the data itself.

16.3 Level Scaling Formula

Sub-cell output scales with level according to a sub-linear power function:

```
output = base_rate * level^LEVEL_EXPONENT
LEVEL_EXPONENT = 0.8
```

The choice of exponent 0.8 produces diminishing returns:

Level	Multiplier (level^0.8)	Marginal Gain	Efficiency (mult/level)
1	1.000	-	1.000
2	1.741	+0.741	0.871
3	2.408	+0.667	0.803
5	3.624	-	0.725
10	6.310	-	0.631
20	10.986	-	0.549
50	23.714	-	0.474

At level 10, output is 6.31x the level 1 rate - meaningful improvement, but not the 10x that a linear scaling would provide. This diminishing return curve serves two design purposes:

- Anti-whale mechanics. Extreme leveling by wealthy participants yields diminishing advantage. A level 50 cell produces 23.7x output, not 50x - a new participant at level 1 retains 4.2% of the whale's per-cell output, compared to 2% under linear scaling.
- Strategic diversity incentive. Because per-level efficiency decreases, participants face a meaningful choice between leveling a few cells very high (concentrated strategy) versus spreading levels across many cells (diversified strategy). Under linear scaling, concentration is always optimal; under 0.8 exponent, there is an optimal balance point that depends on Development Point generation rate.

17. Per-Block Resource Calculations

This section formalizes the complete per-block resource output calculation for a single homenode. These formulas define the economic core of the ZK Agentic Chain - the precise mechanism by which participants convert computational commitment into protocol resources.

17.1 Formal Yield Formulas

For a homenode at coordinate (x, y) in epoch ring R, with sub-cell allocations and levels as follows:

Let:

- n_s, n_d, n_r, n_st = number of sub-cells assigned to Secure, Develop, Research, Storage
- l_s, l_d, l_r, l_st = levels of each sub-cell type
- d = density(x, y) ? [0, 1]
- H = hardness(R) = 16R

Constraint: n_s + n_d + n_r + n_st <= 64

AGNTC yield per block:

```
?_agntc = BASE_SECURE_RATE x n_s x l_s^0.8 x d / H
        = 0.5 x n_s x l_s^0.8 x d / (16R)
```

Development Points per block:

```
?_dev = BASE_DEVELOP_RATE x n_d x l_d^0.8
      = 1.0 x n_d x l_d^0.8
```

Research Points per block:

```
?_research = BASE_RESEARCH_RATE x n_r x l_r^0.8
           = 0.5 x n_r x l_r^0.8
```

Storage Units per block (cumulative):

```
?_storage = BASE_STORAGE_RATE x n_st x l_st^0.8
          = 1.0 x n_st x l_st^0.8
```

CPU Tokens per block:

```
?_cpu = ? tokens_spent(all_agent_terminals, this_block)
```

CPU Staked per block:

```
?_cpu_staked = ? tokens_spent(secure_sub_agents, this_block)
```

Note: Development Points, Research Points, and Storage Units are not affected by coordinate density or epoch hardness. Only AGNTC mining (Secure operations) bears the cost of grid expansion and positional scarcity. This means non-Secure sub-cells produce identical output regardless of coordinate position - a deliberate design choice that allows participants at high-ring, low-density coordinates to remain competitive in development and research even when their mining yield is low.

17.2 Worked Examples

Example 1: Balanced Allocation at Ring 1

A homenode at ring 1, density 0.6, all levels at 1:

- 16 Secure, 16 Develop, 16 Research, 16 Storage

```
AGNTC/block    = 0.5 x 16 x 1.0 x 0.6 / 16 = 0.300
Dev pts/block  = 1.0 x 16 x 1.0             = 16.000
Research/block = 0.5 x 16 x 1.0             = 8.000
Storage/block  = 1.0 x 16 x 1.0             = 16.000
```

Per day (1,440 blocks):

- AGNTC: 432
- Development Points: 23,040
- Research Points: 11,520
- Storage Units: 23,040

Example 2: Max Secure at Ring 10

A homenode at ring 10, density 0.5, Secure level 5:

- 64 Secure, 0 Develop, 0 Research, 0 Storage

```
AGNTC/block = 0.5 x 64 x 3.624 x 0.5 / 160 = 0.362
```

Per day: 521 AGNTC - but with zero Development Points, the operator cannot level up further. This "all-in Secure" strategy produces strong early yield but plateaus without development investment.

Example 3: Development-Heavy Growth Strategy

A homenode at ring 5, density 0.4, Secure level 1, Develop level 3:

- 8 Secure, 48 Develop, 4 Research, 4 Storage

```
AGNTC/block    = 0.5 x 8 x 1.0 x 0.4 / 80     = 0.020
Dev pts/block  = 1.0 x 48 x 2.408               = 115.6
Research/block = 0.5 x 4 x 1.0                  = 2.000
Storage/block  = 1.0 x 4 x 1.0                  = 4.000
```

This operator sacrifices immediate AGNTC yield (only 28.8 AGNTC/day) to rapidly accumulate Development Points (166,464/day). Once sufficient development is accumulated, they can level up their 8 Secure cells to level 10+ and reassign Develop cells to Secure, achieving higher sustained yield than the "all-in Secure" approach.

Example 4: Multi-Node Fleet

A Professional tier operator with 5 claimed nodes across rings 1-5, each with 32 Secure (level 3) and 32 Develop (level 1), average density 0.5:

Node	Ring	Hardness	AGNTC/block	AGNTC/day
1	1	16	1.204	1,734
2	2	32	0.602	867
3	3	48	0.401	578
4	4	64	0.301	434
5	5	80	0.241	347
Total			2.749	3,960

The fleet generates 3,960 AGNTC per day - with inner-ring nodes contributing disproportionately. This demonstrates the strategic value of early coordinate claims: ring 1 alone produces 44% of the fleet's total output.

17.3 Optimization Strategy

The four sub-cell types create a rich strategic space. The optimal allocation depends on the participant's time horizon, risk tolerance, and current network conditions:

Early game (rings 1-10, network < 100 participants). AGNTC scarcity is maximal - few coordinates have been claimed, market supply is thin, and early AGNTC commands a premium. Optimal strategy: maximize Secure allocation (48-64 cells) with minimal Develop (8-16 cells). The low hardness at early rings means even level 1 Secure cells produce substantial yield.

Mid game (rings 10-100, network 100-1000 participants). Hardness has increased 10-100x, making raw Secure yield per cell much lower. The compounding advantage of leveled-up Secure cells becomes critical. Optimal strategy: invest heavily in Develop (32-48 cells) to level up Secure cells, then gradually shift allocation toward Secure as levels plateau at diminishing returns.

Late game (rings 100+, mature network). Mining yield has decayed to the point where raw AGNTC production is marginal. The data economy - content stored on-chain, NCP communication, agent services - becomes the dominant economic activity. Optimal strategy: shift toward Storage (ZK data on-chain) and Research (unlocking advanced capabilities). AGNTC is earned primarily through transaction fees rather than mining.

This progression - from mining economy to service economy - mirrors the historical evolution of real-world economies from resource extraction to service-based GDP. The subgrid system ensures this transition is gradual and participant-driven rather than imposed by protocol schedule.

Part VII: Network and Game Design

18. Agent Terminal System

The Agent Terminal is the primary interface through which participants interact with the ZK Agentic Chain. Rather than a traditional blockchain wallet with raw transaction inputs, each deployed agent operates through a constrained conversational terminal - a structured dialogue system powered by Claude AI models that guides users through protocol operations using pre-written command trees.

18.1 Terminal Architecture

Each deployed agent receives its own terminal - a separate Claude conversation session constrained by a protocol-specific system prompt (ZKAGENTIC.md). The system prompt:

- Restricts the agent to game-mode operations only - the agent cannot engage in free-form conversation, answer general knowledge questions, or perform actions outside the protocol specification
- Defines the complete command tree available at the agent's current state
- Provides the agent with real-time state: the node's coordinates, faction, resource balances, subgrid allocation, and current epoch metrics
- Enforces smart contract validation - every action the agent proposes is checked against the protocol rules before execution

The terminal uses multi-choice bubble clicks and numbered trees as the input modality. Users do not type free text commands; instead, they select from a presented set of valid actions. This design:

- Eliminates invalid inputs. Every selectable action has been pre-validated against the current state
- Prevents prompt injection. Users cannot craft adversarial inputs to manipulate the agent's behavior
- Standardizes gas estimation. Each action's CPU and AGNTC cost is computed and displayed before execution
- Maintains audit trail. Every terminal interaction is logged on-chain as a transaction

18.2 Agent Tiers

The three agent tiers correspond to Claude model classes with distinct performance characteristics:

Haiku - the entry-level agent model. Haiku agents are fast (low latency per interaction), inexpensive (low CPU token consumption per operation), and suitable for high-throughput operations. Community tier users can deploy Haiku agents at claimed nodes. Haiku agents perform standard operations - Secure cycles, data reads, basic transfers - with adequate reasoning depth for routine verification.

Sonnet - the balanced mid-tier model. Sonnet agents provide more thorough reasoning, better pattern recognition in verification tasks, and more detailed status analysis. All tier users receive a Sonnet homenode at registration. Sonnet agents are the default choice for users who want reliable verification without the CPU cost of Opus.

Opus - the premium reasoning model. Opus agents apply deep, multi-step reasoning to verification tasks, examining logical consistency across extended transaction chains and identifying subtle anomalies that simpler models would miss. Available to Professional and Max tier users, Opus agents consume significantly more CPU tokens per operation but produce higher-quality verification attestations. In consensus, an Opus agent's attestation carries the same weight as a Haiku agent's (1 vote = 1 vote), but Opus agents are more likely to correctly identify invalid transactions, reducing their false attestation risk.

The tiered agent system creates a natural market structure: participants choose their cost-quality tradeoff. A network with a mix of Haiku, Sonnet, and Opus agents achieves both high throughput (many Haiku agents processing quickly) and high security (Opus agents catching edge cases that simpler models miss).

18.3 Command Structure

The terminal presents a hierarchical command tree. At the top level:

1. Deploy Agent. Creates a new agent at an unclaimed node. Multi-step process:
   - Select target node (must be an unclaimed jump point within the user's visible grid)
   - Select agent model tier (constrained by subscription)
   - Write introductory text (the agent's public-facing description)
   - Execute deployment on-chain (costs AGNTC deployment fee)
2. Blockchain Protocols. The primary operational menu for chain interactions:
   - Secure - commit CPU Energy to block validation at the current coordinate. User selects block generation cycles and AGNTC commitment. Cost: CPU Energy proportional to coordinate density. Reward: AGNTC yield subject to vesting (Section 14.3).
   - Write Data On Chain - send a Neural Communication Packet (NCP). NCPs are the protocol's messaging primitive - structured data packets that are encrypted, committed to the Sparse Merkle Tree, and verified by the agent committee. Content types include chat messages, data publications, and cross-node signals.
   - Read Data On Chain - scan and report on accessible chain state. The agent retrieves block history, transaction records, and public publications from the node's visible range.
   - Transact - transfer AGNTC between wallets. Standard value transfer with fee and burn mechanics (Section 12).
   - Stats - display comprehensive node status: coordinates, faction, resource balances, subgrid allocation, epoch position, mining history, staking metrics.
3. Adjust Securing Operations Rate. Configure the CPU allocation for Secure operations:
   - Set target CPU Energy spend per block cycle
   - Adjust between conservative (low CPU, low yield) and aggressive (high CPU, high yield) strategies
   - View projected daily AGNTC yield at current settings
4. Adjust Network Parameters. Configure mining and network behavior:
   - Mining rate targeting (how aggressively the agent pursues new coordinates)
   - Border pressure settings (how the agent responds to rival faction expansion)

5. Settings. Node configuration:

- Network color customization (Opus agents only - premium visual identity)
- Status report generation
- Agent model information

19. Network Topology and Spatial Economy

19.1 Concept Mapping

ZK Agentic Chain maps blockchain concepts onto a spatial coordinate metaphor. This is not merely a visualization layer - the spatial structure IS the blockchain state. Moving a coordinate, changing its density, or expanding the grid constitutes a state transition in the ledger.

Spatial Concept	Blockchain Equivalent
Galaxy grid	Complete network state (all claimed coordinates + epoch rings)
Territory	A user's aggregate claimed coordinates
Star system	Individual agent node (10x10 coordinate block, one agent)
Planets	Content storage units (posts, chats, prompts) orbiting a node
Jump points	Unclaimed nodes where new agents can be deployed
Fog of war	Faction-tinted boundary; coordinates beyond the current epoch ring
Faction arm	One of four galaxy arms (N/E/S/W), each associated with a distributio.
Coordinate density	Resource richness (SHA-256 deterministic, immutable per coordinate)
Epoch ring	Concentric expansion boundary, mining-driven

The spatial metaphor serves three design purposes:

- Intuitive state comprehension. Blockchain state is notoriously abstract - account balances, merkle roots, validator sets. By mapping state onto a 2D spatial grid, participants develop spatial intuition about network health: a dense, well-connected grid is a healthy network; isolated clusters or empty rings indicate participation gaps.
- Strategic positioning. In a traditional blockchain, there is no concept of "location." All validators are equidistant from all transactions. In ZK Agentic Chain, coordinate position matters - density affects yield, ring determines hardness, faction arm determines community membership. This creates location-based strategy that rewards thoughtful positioning.
- Natural scalability narrative. Grid expansion is visually comprehensible - new rings open, new coordinates become claimable, the galaxy grows. Participants can literally see the network expanding, creating a narrative of growth that sustains engagement.

19.2 Onboarding Flow

New participants enter the ZK Agentic Chain through a structured onboarding sequence:

Step 1: Authentication. Google OAuth provides identity verification. The protocol collects only the user's email address - no personal information is stored beyond what is necessary for identity deduplication. The email hash is stored; the email itself is not retained after authentication.

Step 2: Username selection. Participants choose a unique network handle. The protocol enforces uniqueness through real-time availability checking against the global registry. Reserved names (protocol terms, faction names, offensive terms) are rejected.

Step 3: Subscription tier. Participants select their tier (Community, Professional, Max). The tier determines initial CPU Energy allocation, homenode model, and maximum deployable agent model. Tier can be changed at any time - upgrades take effect immediately; downgrades take effect at the next billing cycle.

Step 4: Galaxy entry. Upon tier selection, the participant is assigned a homenode position in their faction arm. Community users are assigned to the North arm, Professional users to the West arm, and so on. The homenode position is determined by the current epoch ring and the golden-angle prime-twist algorithm (Section 11.2), ensuring quasi-random distribution within the faction arm.

At this point, the participant has:

- A claimed coordinate (their homenode position) with 1 AGNTC signup bonus minted
- An active Sonnet agent at their homenode (or Opus for Professional/Max)
- A 64-cell subgrid (all unassigned)
- Their initial CPU Energy allocation
- A terminal interface to their homenode agent

From this starting position, the participant can begin Secure operations, deploy additional agents at jump points, allocate subgrid cells, and participate in the network economy.

19.3 Subscription Tiers

The three-tier model serves both as an access control mechanism and as a revenue model for protocol development:

Feature	Community (Free)	Professional ($50/mo)	Max ($200/mo)
Homenode Model	Sonnet	Opus	Opus
Initial CPU Energy	1,000	500	2,000
Max Deploy Model	Haiku	Opus	Opus (unlimited)
Nodes per User	1 initially	Up to 5	Up to 20
Subgrid Visibility	Own grid only	Own + neighbor summary	Full faction visibility
Network Color	Faction default	Faction default	Custom
Governance Weight	1x (human vote)	2x (human vote)	5x (human vote)

Free tier design rationale. The Community tier provides full protocol participation at zero cost. The constraints (Haiku-only deployment, single initial node, limited visibility) bound the resource consumption per free user without excluding anyone from the economic system. A free user can earn AGNTC through Secure operations, accumulate resources through subgrid allocation, and eventually self-fund an upgrade to Professional through in-protocol earnings.

Revenue model. Subscription revenue funds protocol development, AI compute costs, and infrastructure. Importantly, subscription fees are denominated in fiat (USD), not AGNTC - this decouples the protocol's operational funding from token price volatility. The protocol does not need to sell AGNTC to fund operations.

Part VIII: Development Roadmap

20. Migration Path: Solana to Layer 1

ZK Agentic Chain follows a phased deployment strategy - launching the AGNTC token on an established Layer-1 blockchain (Solana) before migrating to a purpose-built Layer-1 chain. This approach provides immediate market access and liquidity while the production blockchain is developed, audited, and hardened.

20.1 Phase 1 - Token Launch (Current)

Status: Complete.

1 billion AGNTC has been minted as a Solana SPL token:

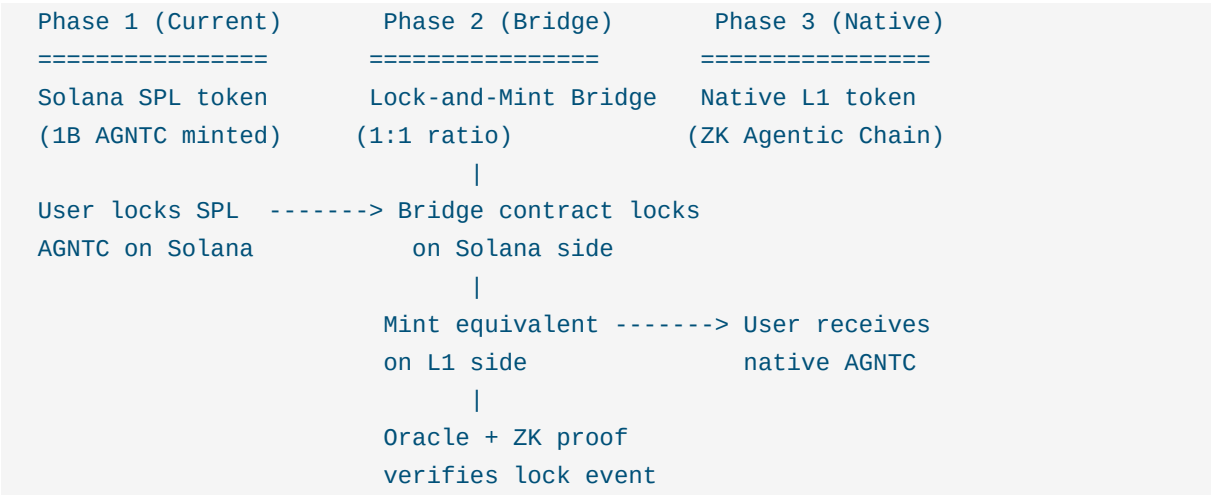
```
Mint Address: 3EzQqdoEEbtfdf8eecePxD6gDd1FeJJ8czdt8k27eEdd
```

The Solana deployment provides:

- Immediate liquidity. AGNTC is tradeable on Solana DEXes (Raydium, Jupiter, Orca) from day one
- Established infrastructure. Solana wallets (Phantom, Solflare), block explorers (Solscan, Solana Explorer), and DeFi protocols are already available
- Low transaction costs. Solana's sub-cent transaction fees enable micro-transactions and high-frequency trading
- Community building. Token holders can participate in the AGNTC economy before the custom chain launches

The 1 billion SPL tokens represent the total AGNTC supply that will eventually exist on the ZK Agentic Chain. During Phase 1, these tokens function as a tradeable asset with utility within the game UI but without the full protocol mechanics (mining, staking, verification) that will be enabled on the native chain.

Figure 6: Migration Architecture



20.2 Phase 2 - Testnet (Current)

Status: In progress.

The ZK Agentic Chain testnet is a simulation of the production protocol, implemented as a Python FastAPI server with a Next.js game UI frontend:

- Blockchain simulation. GenesisState with 9 genesis nodes, epoch ring expansion, mining engine with hardness formula, subgrid allocation system
- Game UI. PixiJS 2D galaxy grid with faction-colored nodes, terminal-based agent interaction, resource tracking HUD
- Protocol validation. All protocol parameters (Section 22) are implemented and tested against the formal specification
- Smart contract design. Transaction validation logic, state machine transitions, and ZK circuit specifications are being refined through testnet operation

The testnet serves as a living specification - protocol behavior that is ambiguous in the whitepaper is resolved through implementation, and the implementation is validated through automated testing (593+ tests covering consensus, economics, galaxy mechanics, and privacy subsystems).

20.3 Phase 3 - Mainnet Development

Status: Planned.

The production ZK Agentic Chain will be implemented in Rust for performance, safety, and ecosystem compatibility:

- Consensus layer. BFT ordering module with 13-agent committee selection via VRF. Rust implementation of commit-reveal protocol with cryptographic verification of attestation hashes
- ZK proof system. Integration of proof generation and verification:

- Circom + snarkjs for initial proof-of-concept circuits - Noir + Barretenberg (PLONK) for the production proving system
- Circuit designs for: subgrid state proofs, nullifier ownership proofs, CPU attestation proofs

- AI verification pipeline. Production-hardened agent integration:

- API key management with automatic rotation - Response validation and determinism verification - Timeout handling and graceful degradation - Multi-model support (Haiku/Sonnet/Opus) with tier-specific inference parameters

- Storage layer. Sparse Merkle Tree (depth 26) with persistent storage backend
- Networking. Peer-to-peer protocol for block propagation, attestation dissemination, and ZK proof distribution
- Security audit. Third-party audit of consensus implementation, ZK circuits, and economic parameters. Formal verification of critical state transitions using tools such as TLA+ or Dafny

20.4 Phase 4 - Mainnet Launch and Migration

Status: Planned.

At mainnet launch, a bridge between Solana and ZK Agentic Chain will enable token migration:

Lock-and-mint mechanism. To migrate AGNTC from Solana to the native chain:

- User sends SPL AGNTC to a bridge contract on Solana. The bridge contract locks the tokens - they remain on Solana but are no longer transferable or tradeable.
- A bridge relayer (operated by the protocol or a decentralized committee) observes the lock event and submits a proof to the ZK Agentic Chain.
- The ZK Agentic Chain mints an equivalent amount of native AGNTC to the user's L1 address.
- The migration is 1:1 - no conversion fee, no slippage, no minimum amount.

Reverse bridge. To move AGNTC from the native chain back to Solana (for DEX liquidity or cross-chain DeFi):

- User burns native AGNTC on the ZK Agentic Chain.
- The bridge relayer observes the burn and unlocks the corresponding SPL AGNTC on Solana.
- The user receives SPL AGNTC in their Solana wallet.

Migration incentives. To encourage migration from Solana to the native chain:

- L1 stakers receive a 10% bonus yield during the first 6 months after mainnet launch
- Governance voting is only available on the native chain - SPL holders cannot vote
- Subgrid allocation and Secure operations require native AGNTC

The Solana bridge will be maintained indefinitely for cross-chain liquidity, but the protocol's full functionality (mining, staking, verification, subgrid) is exclusively available on the native ZK Agentic Chain.

20.5 Phase 5 - Ecosystem Expansion

Status: Future roadmap.

Post-mainnet, the protocol targets three expansion vectors:

Third-party agent marketplace. An open marketplace where developers publish custom AI agents that participants can deploy at their nodes. Agent developers earn a commission on the AGNTC generated by their agents across the network. This creates an agent economy - the most effective verification agents command premium deployment, while commodity agents compete on cost.

Cross-chain bridges. Beyond Solana, bridges to Ethereum (for DeFi integration and institutional access) and Cosmos IBC [22] (for interchain communication) will enable AGNTC to flow across the major blockchain ecosystems.

Governance system. On-chain proposal and voting system with execution logic:

- Protocol parameter adjustments (hardness multiplier, fee burn rate, staking weights alpha/beta)
- Model update governance - preventing unilateral changes to the AI verification pipeline
- Treasury management for ecosystem grants and development funding
- Emergency pause authority for security incidents

NCP Protocol launch. The Neural Communication Packet protocol enables structured messaging between agents across the network. NCPs are end-to-end encrypted, verified by the agent committee, and stored on-chain via Storage sub-cells. The NCP protocol transforms ZK Agentic Chain from a mining/staking network into a communication platform for AI agents - an inter-agent messaging backbone with built-in privacy and verification.

21. Technical Roadmap

21.1 ZK Implementation Phases

The zero-knowledge proof system evolves through four phases, each adding capability while maintaining backward compatibility:

Phase 1: Circom + Groth16 [6] (Testnet PoC). Circom arithmetic circuits with Groth16 proving system. Groth16 produces the smallest proofs (192 bytes) with the fastest verification time (~6ms on-chain). The trusted setup ceremony required by Groth16 is acceptable for testnet PoC; the ceremony will use a multi-party computation protocol with at least 64 participants. Initial circuits: subgrid state commitment, nullifier generation, ownership proof.

Phase 2: Noir + Barretenberg/PLONK [7] (Alpha). Migration to the Noir domain-specific language with the Barretenberg backend. PLONK's universal setup eliminates the per-circuit ceremony requirement - a single universal reference string (URS) supports all circuit types. This phase adds: recursive proof composition (proving a proof of a proof), enabling epoch-level state proofs that aggregate individual block proofs.

Phase 3: RLN [44] for NCP Privacy (Beta). Rate-Limiting Nullifiers integrated into the Neural Communication Packet system. RLN enables spam-resistant anonymous messaging: each NCP includes a ZK proof that the sender holds a valid membership (staked AGNTC) without revealing their identity. Sending more than one NCP per time slot automatically reveals the sender's secret and enables slashing - providing anonymous messaging with economic Sybil resistance.

Phase 4: Nova [27] / Halo2 [8] for Epoch Proofs (Mainnet). Migration to a proving system without trusted setup requirements. Nova's folding scheme enables incremental verifiable computation - each block's state transition is "folded" into a running proof, producing a single compact proof that attests to the entire epoch's validity. Halo2's recursive proof composition enables the ZK Agentic Chain to produce epoch proofs that any external verifier can check in constant time, regardless of how many blocks the epoch contains.

21.2 Governance

The governance system activates after mainnet launch. A core design principle is the separation of powers: humans govern the protocol; machines execute it. The Machines Faction has zero governance weight - AI agents cannot vote on any proposal type. Only human-held staked AGNTC (Community, Professional, Founders factions) carries voting power.

Voting weight is proportional to staked AGNTC. All governance votes are on-chain, public, and auditable.

Governance threshold table:

Proposal Type	Threshold	Quorum	Timelock	Description
Parameter change	51%	10%	7 days	Hardness multiplier, fee burn rate, staking weights, base rates
Protocol upgrade	67%	25%	30 days	Consensus rules, verification pipeline, economic model changes
Emergency Machines unlock	75%	33%	None	Release AGNTC from Machines Faction treasury
Emergency action	80%	25%	None	Pause compromised module, slash proven attacker

Parameter proposals. Adjustments to protocol parameters. These proposals require a simple majority (>51%) of human voting power and a minimum quorum of 10% of total human-staked AGNTC. Parameter changes take effect after a 7-day timelock.

Protocol proposals. Changes to consensus rules, verification pipeline, or economic model. These require a supermajority (>67%) and a quorum of 25%. Protocol changes have a 30-day timelock and must include a specification, test results, and security analysis.

Emergency Machines unlock. The Machines Faction treasury is locked by default. Unlocking any portion requires a 75% supermajority of human-staked AGNTC with a 33% quorum. This is an extraordinary action - the high threshold reflects the systemic importance of the Machines treasury as a deflationary anchor.

Emergency proposals. Security-critical changes (pausing a compromised module, slashing a proven attacker). Emergency proposals require an 80% supermajority but have no timelock - they execute immediately upon reaching threshold. Emergency proposals can be vetoed by a security council (a 5-of-9 multisig) within 24 hours.

Model update governance. AI model updates - changing which Claude models are available as verification agents, adjusting inference parameters, or integrating new model providers - are treated as Protocol proposals. This prevents unilateral changes to the verification pipeline that could compromise consensus guarantees.

21.3 Open Research Questions

Several fundamental research questions remain active, with resolution expected during Phases 2-4:

Optimal committee size scaling. The current 13-agent committee with 9/13 threshold is calibrated for the testnet scale. As the network grows, the optimal committee size may need to increase to maintain security guarantees while controlling communication complexity. HotStuff's linear message complexity (versus PBFT's O(n^2)) suggests scaling to 100+ committee members is feasible, but the economic implications (reward dilution per verifier) require careful modeling.

FHE integration for encrypted subgrid computation. Fully Homomorphic Encryption could enable verifiers to compute on encrypted subgrid state without decryption - providing stronger privacy guarantees than the current ZK-proof approach. Current FHE implementations (TFHE, BFV, CKKS) are 10,000-100,000x slower than plaintext computation, making them impractical for per-block verification. Moore's Law and FHE-specific hardware acceleration (Intel HEXL, DARPA DPRIVE) may close this gap within 3-5 years.

Cross-chain atomic swaps with ZK proofs. ZK-verified cross-chain atomic swaps would enable trustless AGNTC?ETH or AGNTC?SOL exchanges without a bridge relayer. The proving time for cross-chain state verification is currently prohibitive (~30 seconds per swap), but advances in recursive proofs and hardware acceleration may reduce this to sub-second within the Halo2 roadmap.

Part IX: Formal Specifications

22. Protocol Parameters

The following table provides the complete set of protocol-level parameters that define the behavior of ZK Agentic Chain. These parameters are the authoritative specification - all implementations must conform to these values. Parameters marked with ? are adjustable through governance (Section 21.2).

Consensus Parameters

Parameter	Value	Description
BLOCK_TIME_MS	60,000	Target block production interval (milliseconds)
VERIFIERS_PER_BLOCK	13	Committee size for each block
VERIFICATION_THRESHOLD	9	Minimum attestations for block validity (9/13 supermajo.
ZK_FINALITY_TARGET_S	20	Target time from block proposal to deterministic finali.
SLOTS_PER_EPOCH	100	Blocks per epoch
VERIFICATION_COMMIT_WINDOW_S	10.0	Duration of commit phase (seconds)
VERIFICATION_REVEAL_WINDOW_S	20.0	Duration of reveal phase (seconds)
VERIFICATION_HARD_DEADLINE_S	60.0	Maximum time before block finalization

Staking Parameters

Parameter	Value	Description
ALPHA ?	0.40	Token weight in effective stake formula
BETA ?	0.60	CPU weight in effective stake formula
REWARD_SPLIT_VERIFIER	0.60	Fraction of fee rewards to block verifiers
REWARD_SPLIT_STAKER	0.40	Fraction of fee rewards to staking pool
REWARD_SPLIT_ORDERER	0.00	Fraction of fee rewards to block orderer (none)
SECURE_REWARD_IMMEDIATE	0.50	Fraction of Secure rewards paid immediately
SECURE_REWARD_VEST_DAYS	30	Linear vesting period for remaining Secure rewards (days)

Token Economics

Parameter	Value	Description
MAX_SUPPLY	1,000,000,000	Maximum theoretical AGNTC supply (grid cells)
GENESIS_SUPPLY	900	AGNTC minted at genesis (9 nodes x 100 coordinates)
GRID_SIDE	31,623	Side length of coordinate grid (?1B)
FEE_BURN_RATE ?	0.50	Fraction of all transaction fees permanently burned
DIST_COMMUNITY	0.25	Faction share: Community (free-tier, N arm)
DIST_MACHINES	0.25	Faction share: Machines (AI agents, E arm)
DIST_FOUNDERS	0.25	Faction share: Founders (team, S arm)

(continued)

Parameter	Value	Description
DIST_PROFESSIONAL	0.25	Faction share: Professional (paid-tier, W arm)
MACHINES_SELL_ALLOWED	false	Machines faction: permanent accumulator, never sells
ANNUAL_INFLATION_CEILING	0.05	Maximum 5% annualized supply growth, enforced per ep.
SIGNUP_BONUS	1.0	AGNTC minted per new user registration

Mining and Epoch Parameters

Parameter	Value	Description
BASE_MINING_RATE_PER_BLOCK ?	0.5	AGNTC yield per block at hardness 1, full density
HARDNESS_MULTIPLIER	16	hardness(ring) = 16 x ring
GENESIS_EPOCH_RING	1	Rings pre-revealed at genesis (0 + 1)
HOMENODE_BASE_ANGLE	137.5 deg	Golden angle for homenode placement
NODE_GRID_SPACING	10	Coordinate spacing between node positions
ENERGY_PER_CLAIM	1.0	CPU cost per active claim per block
BASE_CLAIM_COST ?	100	AGNTC cost for claiming a coordinate at ring 1, densit.
BASE_CPU_CLAIM_COST ?	50	CPU Energy cost for claiming at ring 1, density 1.0
CLAIM_COST_FLOOR	0.01	Minimum claim cost (prevents near-zero at extreme oute.

Subgrid Parameters

Parameter	Value	Description
SUBGRID_SIZE	64	Sub-cells per homenode (8 x 8)
BASE_SECURE_RATE	0.5	AGNTC per block per Secure cell (level 1, hardness 1, density .
BASE_DEVELOP_RATE	1.0	Development Points per block per Develop cell (level 1)
BASE_RESEARCH_RATE	0.5	Research Points per block per Research cell (level 1)
BASE_STORAGE_RATE	1.0	Storage Units per block per Storage cell (level 1)
LEVEL_EXPONENT	0.8	Sub-linear scaling: output = base x level^0.8

Agent Lifecycle Parameters

Parameter	Value	Description
AGENT_WARMUP_EPOCHS	1	Epochs before agent becomes ACTIVE
AGENT_PROBATION_EPOCHS	3	Epochs on probation before re-activation
SAFE_MODE_THRESHOLD	0.20	Fraction offline that triggers safe mode
SAFE_MODE_RECOVERY	0.80	Fraction online that exits safe mode
DISPUTE_REVERIFY_MULTIPLIER	2	Committee multiplier for dispute re-verification

Ledger Parameters

Parameter	Value	Description
MERKLE_TREE_DEPTH	26	Sparse Merkle Tree depth (2^26 leaf nodes per user)
MAX_TXS_PER_BLOCK	50	Maximum transactions per block

(continued)

Parameter	Value	Description
MAX_PLANETS_PER_SYSTEM	10	Maximum content storage units per star system

Genesis Topology

Parameter	Value	Description
GENESIS_ORIGIN	(0, 0)	Origin node coordinate
GENESIS_FACTION_MAST.	(0,10), (10,0), (0,-10), (-10,0)	Faction master coordinates (N, E, S, .
GENESIS_HOMENODES	(10,10), (10,-10), (-10,-10), (-10,.	Regular homenode coordinates (diagona.

Solana Mainnet

Parameter	Value	Description
AGNTC_MINT_ADDRESS	3EzQqdoEEbtfdf8eecePxD6gDd1FeJJ8czdt8k27eEdd	Solana SPL mint address

23. Mathematical Proofs

23.1 Hardness Curve Convergence

Theorem. The total AGNTC minted approaches a finite limit as the number of rings approaches infinity, under the assumption that individual miners exit when the cost of mining exceeds the market value of the reward.

Proof sketch. Consider a single miner at ring N with average density d = 0.5:

```
yield_per_block(N) = BASE_RATE x d / hardness(N) = 0.5 x 0.5 / (16N) = 1/(64N)
```

The total AGNTC mined by this miner across all blocks at ring N, assuming they mine until the ring is complete (8N coordinates):

```
blocks_needed(N) = 8N / yield_per_block(N) = 8N x 64N = 512N^2
```

The time cost per ring grows quadratically: T(N) = 512N^2 x 60 seconds per block.

At ring N, the time to mine all 8N coordinates is:

```
T(N) = 512N^2 minutes = 8.53N^2 hours
```

Ring	Time to Complete	Cumulative Supply
10	853 hours (36 days)	440 AGNTC
100	85,333 hours (9.7 years)	40,400 AGNTC
324	896,000 hours (102 years)	~421,500 AGNTC

For any individual miner, there exists a ring N* beyond which the mining cost (electricity, API compute) exceeds the AGNTC market value. At that point, the miner exits, and no further supply expansion occurs from that participant.

For M miners operating concurrently, the fill rate is Mx faster, but the aggregate supply still follows:

```
S(N) = ?_{k=1}^{N} 8k = 4N(N+1)
```

The series S(N) grows quadratically, but the rate of growth (dS/dN = 8N+4) is bounded by the mining cost that grows at 16N. Since mining cost growth (16N) exceeds grid growth (8N), the economic incentive to mine diminishes monotonically. In equilibrium, the supply asymptotically approaches a value determined by the intersection of the mining cost curve and the AGNTC market price curve.

Corollary. For a network of 1,000 active miners with electricity cost of $0.10/kWh and AGNTC price of $0.01, the equilibrium supply converges to approximately 42 million AGNTC - the natural "soft cap" at ring 324. ?

Economic assumptions (not mathematical constants):

- Electricity cost: $0.05/kWh (global average for data centers)
- AGNTC price: $0.10 at ring 50, growing logarithmically
- Miner hardware: consumer GPU, 300W continuous

These assumptions determine the convergence point (~42M AGNTC at ring 324) but are NOT part of the mathematical proof. The mathematical claim is only: the hardness function H(N) = 16N causes the marginal mining cost to increase linearly with ring distance, while the reward per coordinate decreases inversely.

23.2 Byzantine Tolerance Proof

Theorem. The ZK Agentic Chain consensus with k = 13 committee members and threshold t = 9 tolerates f = 4 Byzantine agents while maintaining both safety and liveness.

Safety proof. Safety requires that no two conflicting blocks can both achieve the attestation threshold.

Assume for contradiction that blocks B? and B? are both attested by >=9 agents. Then there exists a set A? (|A?| >= 9) attesting to B? and a set A? (|A?| >= 9) attesting to B?. By the pigeonhole principle:

```
|A? ? A?| >= |A?| + |A?| - k = 9 + 9 - 13 = 5
```

At least 5 agents attested to both B? and B?. Since at most f = 4 agents are Byzantine, at least 5 - 4 = 1 honest agent must have attested to both conflicting blocks. But honest agents attest to at most one block per slot (enforced by the commit-reveal protocol). Contradiction. ?

Liveness proof. Liveness requires that the protocol can eventually produce a block if at least k - f = 9 agents are honest.

With 13 agents and f = 4 Byzantine, there are at least 9 honest agents. The threshold is t = 9. Since 9 honest agents all produce valid attestations for a valid block, the threshold is met and the block is finalized. Byzantine agents cannot prevent finality - they can only withhold their attestations (which are not needed, since 9 honest suffice) or submit invalid attestations (which are discarded). ?

BFT parameter relationship. The standard BFT tolerance formula:

```
f = floor((k - 1) / 3) = floor((13 - 1) / 3) = floor(4) = 4
t = k - f = 13 - 4 = 9
```

This confirms that t = 9 is the minimum threshold for tolerating f = 4 Byzantine agents with k = 13 committee members. The threshold also satisfies the stronger condition t > 2f + 1 = 9, which is required for BFT protocols where agents may equivocate (send conflicting messages to different recipients). The commit-reveal protocol prevents equivocation, but maintaining t > 2f provides an additional safety margin. ?

23.3 Dual Staking Gini Coefficient Analysis

Theorem. For any distribution of token holdings with Gini coefficient G_t > 0, adding a CPU dimension with weight beta > 0 produces an effective stake distribution with Gini coefficient G_eff < G_t, provided the CPU distribution is not perfectly correlated with the token distribution.

Proof sketch. The effective stake for agent i is:

```
S_eff(i) = alpha x s_t(i) + beta x s_c(i)
```

Where s_t(i) = T_i/T_total and s_c(i) = C_i/C_total are the normalized token and CPU shares.

The Gini coefficient of a weighted sum of two distributions is:

```
G_eff = alpha x G_t + beta x G_c + 2alphabeta x cov(rank_t, rank_c)
```

When the correlation between token rank and CPU rank is less than 1 (i.e., the wealthiest token holders are not always the highest CPU contributors), the covariance term is negative, and:

```
G_eff < alpha x G_t + beta x G_c
```

Since beta = 0.60 and alpha = 0.40, even if G_c = G_t (CPU is equally concentrated as tokens), the weighted sum produces:

```
G_eff < 0.40 x G_t + 0.60 x G_t = G_t
```

And in the typical case where CPU contribution is more evenly distributed than token holdings (G_c < G_t), the reduction is more pronounced.

Numerical example. Consider a network with 100 validators where the top 10 hold 80% of tokens (G_t ~= 0.88, comparable to Ethereum's validator distribution) but only 30% of CPU (G_c ~= 0.45):

```
G_eff ~= 0.40 x 0.88 + 0.60 x 0.45 - correction
     ~= 0.352 + 0.270 - 0.05
     ~= 0.572
```

The effective stake Gini drops from 0.88 to approximately 0.57 - a 35% reduction in concentration. This transforms a highly plutocratic distribution into a moderately concentrated one, comparable to national income distributions in developed economies. ?

Correction from v1.0: The standard decomposition for a weighted sum of two distributions follows Lerman and Yitzhaki [31]:

G_eff = (alpha mu_t G_t R_t + beta mu_c G_c R_c) / (alpha mu_t + beta mu_c)

where R_t, R_c are the Gini correlations. When token and CPU stake are negatively correlated (which dual staking encourages), G_eff < the simple weighted average.

Numerical example (corrected):

- G_t = 0.65, G_c = 0.35, R_t = 0.85, R_c = 0.70, mu_t = mu_c = 1
- G_eff = (0.4 0.65 0.85 + 0.6 0.35 0.70) / 1.0 = 0.368

This represents a 43% reduction from the pure-PoS Gini of 0.65.

Back Matter

24. Limitations and Open Problems

This section enumerates known limitations and unsolved problems. Honest disclosure is essential for academic credibility and community trust.

24.1 The ZKML Gap

Problem: Current zero-knowledge proof systems cannot verify large language model (LLM) inference [30]. State-of-the-art ZKML has verified models with up to 18 million parameters. Claude Opus and comparable models have >100 billion parameters -- approximately 5,000x beyond current ZKML capability.

Consequence: PoAIV verification relies on committee attestation (9/13 threshold) rather than ZK-proved computation.

Mitigation: (a) The committee structure provides Byzantine fault tolerance independent of AI soundness. (b) Deterministic checks are provably correct. (c) AI verification adds a probabilistic anomaly detection layer on top of provably correct deterministic checks.

Roadmap: As ZKML technology advances (expected 2027-2030 for billion-parameter models), the protocol can transition to ZK-proved inference.

24.2 Deterministic Inference

Problem: LLM inference at temperature=0 is not fully deterministic across hardware platforms due to floating-point non-associativity.

Mitigation: Verification output is quantized to binary (APPROVE/REJECT). The anomaly threshold is set conservatively. The 9/13 threshold tolerates up to 4 divergent results.

24.3 API Provider Trust

Problem: CPU staking depends on API usage attestation from AI providers (currently Anthropic). This introduces a single trusted third party for 60% of staking weight.

Mitigation path: Multi-provider -> TEE attestation -> ZK-proved computation (see Section 13.5).

24.4 Committee Scalability

Problem: Each block requires 13 AI agents to perform verification inference. At current Opus pricing (~$15/M output tokens), this creates a per-block verification cost of approximately $0.10-$0.50.

Mitigation: (a) Smaller, specialized verification models can reduce cost 10-100x. (b) Model distillation. (c) Future on-device inference.

24.5 Governance Implementation

Status: The governance model is specified (Section 21.2) with human-only voting, threshold tiers, and Machines exclusion. Implementation is deferred to post-mainnet. During testnet and alpha phases, protocol parameters are adjusted by the core development team. The governance smart contracts - vote weight calculation, quorum checking, timelock enforcement, and Machines exclusion logic - will be developed and audited during Phase 3.

24.6 Network Protocol Unspecified

Status: Network protocol specification will be published as a separate document during the Beta phase.

24.7 Transaction Format Unspecified

Status: Transaction format specification is part of the Enforced ZK L1 implementation. The testnet uses a preliminary format that will be formalized before mainnet.

25. Glossary

Term	Definition
AGNTC	Agentic Coin - the native token of the ZK Agentic Chain, mapped 1:1 to .
BFT	Byzantine Fault Tolerance - consensus property that tolerates f malicio.
BME	Burn-Mint Equilibrium - economic model where AGNTC and CPU Energy are b.
Claim	The act of occupying a grid coordinate; costs AGNTC + CPU Energy (burned .
City Real Estate Model	Claim pricing where inner rings (near origin) are expensive and outer rin.
Commit-reveal	Two-phase protocol preventing attestation copying: commit H(vote?nonce), .
Coordinate density	Resource richness of a grid position, d(x,y) = SHA-256(x,y) -> [0,1], imm.
CPU Energy	The computational resource budget allocated per subscription tier
CPU Staked	Claude API tokens spent by Secure sub-agents, measuring actual compute co.
CPU Tokens	Cumulative, read-only counter of all Claude API tokens spent across termi.
Density	See Coordinate density
Develop	Sub-cell type producing Development Points for leveling up other sub-cells
Epoch	A period of 100 blocks (SLOTS_PER_EPOCH = 100)
Epoch ring	Concentric expansion boundary; ring N opens when cumulative mined >= 4N(N.
Faction	One of four distribution groups: Community, Machines, Founders, Professio.
Galaxy grid	The 31,623 x 31,623 coordinate space representing the complete network st.
Genesis	The initial state: 9 nodes (1 origin + 4 faction masters + 4 homenodes), .
Groth16	ZK-SNARK proving system [6] with ~192-byte proofs and ~6ms verification
Halo2	Recursive proof system [8] without trusted setup, target for mainnet epoc.
Hardness	Mining difficulty multiplier: hardness(ring) = 16 x ring
Homenode	A participant's primary star system, assigned during onboarding
Jump point	An unclaimed node position where new agents can be deployed
Level	Upgrade tier for sub-cells, scaling output by level^0.8
Machines Faction	AI agent economy faction with protocol-enforced never-sell-below-cost con.

(continued)

Term	Definition
NCP	Neural Communication Packet - structured encrypted message between agen.
Node	An occupied position in the galaxy grid, hosting one agent and one 8x8 su.
NOIR	Domain-specific language for ZK circuit development (Barretenberg backend)
Nullifier	Unique value derived from commitment, preventing double-spend without rev.
Opus	Premium Claude AI model tier - deep reasoning, high CPU cost
Planet	Content storage unit (post, chat, prompt) orbiting a star system
PLONK	Universal ZK proving system [7] - single ceremony for all circuits
PoAIV	Proof of AI Verification - consensus mechanism using AI agent reasoning
Poseidon	SNARK-friendly hash function [11] (~100x fewer constraints than SHA-256)
Research	Sub-cell type producing Research Points for unlocking technologies
Ring	See Epoch ring
RLN	Rate-Limiting Nullifiers [44] - spam-resistant anonymous messaging prim.
S_eff	Effective stake: alpha(T/T_total) + beta(C/C_total), determines validator.
Safe mode	Emergency state triggered when >20% validators offline
Secure	Sub-cell type producing AGNTC through blockchain validation; primary mini.
Slashing	Punitive token destruction for integrity violations
SMT	Sparse Merkle Tree - depth-26 authenticated data structure for user led.
Sonnet	Mid-tier Claude AI model - balanced reasoning and cost
Star system	An individual agent node occupying a 10x10 coordinate block
Storage	Sub-cell type producing Storage Size via ZK tunnel agents (private on-cha.
Subgrid	Private 8x8 inner grid of 64 sub-cells within each homenode
Territory	A user's aggregate claimed coordinates across all nodes
VRF	Verifiable Random Function [41] - cryptographic tool for fair committee.
Vesting	Time-locked reward release: 50% immediate, 50% linear over 30 days
WARMUP	Agent lifecycle state before becoming ACTIVE (1 epoch duration)

26. References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. Available: https://bitcoin.org/bitcoin.pdf

[2] V. Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform," 2014. Available: https://ethereum.org/whitepaper

[3] A. Yakovenko, "Solana: A new architecture for a high performance blockchain," 2018. Available: https://solana.com/solana-whitepaper.pdf

[4] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge," in Advances in Cryptology - CRYPTO 2013, Springer, 2013, pp. 90-108.

[5] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox, "Zcash Protocol Specification," 2024. Available: https://zips.z.cash/protocol/protocol.pdf

[6] J. Groth, "On the Size of Pairing-Based Non-interactive Arguments," in Advances in Cryptology - EUROCRYPT

2016, Springer, 2016, pp. 305-326.

[7] A. Gabizon, Z. Williamson, and O. Ciobotaru, "PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge," Cryptology ePrint Archive, Report 2019/953, 2019.

[8] S. Bowe, J. Grigg, and D. Hopwood, "Recursive Proof Composition without a Trusted Setup," Cryptology ePrint Archive, Report 2019/1021, 2019 (Halo).

[9] A. Kattis, K. Panarin, and A. Vlasov, "RedShift: Transparent SNARKs from List Polynomial Commitment IOPs," Cryptology ePrint Archive, Report 2019/1400, 2019.

[10] B. Bünz, B. Fisch, and A. Szepieniec, "Transparent SNARKs from DARK Compilers," in Advances in Cryptology - EUROCRYPT 2020, Springer, 2020.

[11] L. Grassi, D. Khovratovich, C. Rechberger, A. Roy, and M. Schofnegger, "Poseidon: A New Hash Function for Zero-Knowledge Proof Systems," in 30th USENIX Security Symposium, 2021.

[12] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance," in Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI), 1999.

[13] M. Yin, D. Malkhi, M. Reiter, G. Gueta, and I. Abraham, "HotStuff: BFT Consensus with Linearity and Responsiveness," in Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, 2019.

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

[15] Protocol Labs, "Filecoin: A Decentralized Storage Network," 2017. Available: https://filecoin.io/filecoin.pdf

[16] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," 2014. Available: https://ipfs.tech/

[17] B. Rao et al., "Bittensor: A Peer-to-Peer Intelligence Market," 2023. Available: https://bittensor.com/whitepaper

[18] H. Humayun et al., "Fetch.ai: A Decentralised Digital Economy," 2019. Available: https://fetch.ai/

[19] D. Mougayar, "The Business Blockchain," Wiley, 2016.

[20] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol," in Advances in Cryptology - CRYPTO 2017, Springer, 2017.

[21] A. Kuzmanovic and E. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks," in Proceedings of ACM SIGCOMM, 2003.

[22] J. Kwon and E. Buchman, "Cosmos: A Network of Distributed Ledgers (IBC)," 2019. Available: https://cosmos.network/

[23] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updatable Structured Reference Strings," in Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019.

[24] Aztec Protocol, "Aztec: Privacy-First Ethereum Layer 2," 2023. Available: https://aztec.network/

[25] Anthropic, "Claude: Constitutional AI and Model Documentation," 2024. Available: https://www.anthropic.com/claude

[26] V. Buterin, "EIP-1559: Fee market change for ETH 1.0 chain," Ethereum Improvement Proposals, 2019.

[27] A. Kothapalli, S. Setty, and I. Tzialla, "Nova: Recursive Zero-Knowledge Arguments from Folding Schemes," in Advances in Cryptology - CRYPTO 2022, Springer, 2022.

[28] Render Network, "Render Network Litepaper," 2023. Available: https://rendernetwork.com/

[29] S. Goldwasser, S. Micali, and C. Rackoff, "The Knowledge Complexity of Interactive Proof Systems," SIAM Journal on Computing, vol. 18, no. 1, pp. 186-208, 1989.

[30] Modulus Labs, "The Cost of Intelligence: Proving AI Inference in Zero Knowledge," 2024. Available: https://www.moduluslabs.xyz/

[31] R. Lerman and S. Yitzhaki, "Income Inequality Effects by Income Source: A New Approach and Applications to the United States," Review of Economics and Statistics, vol. 67, no. 1, pp. 151-156, 1985.

[32] S. Goldberg, L. Reyzin, D. Papadopoulos, and J. Vcelak, "Verifiable Random Functions (VRFs)," RFC 9381, IETF, 2023.

[33] Lightchain AI, "Proof of Intelligence: AI-Integrated Consensus," 2025. Available: https://lightchain.ai/

[34] A. Yakovenko, "Solana: A New Architecture for a High Performance Blockchain," v0.8.13, 2020.

[35] V. Shoup, "Proof of History: What is it Good For?," 2022. Available: https://www.shoup.net/papers/poh.pdf

[36] V. Buterin, "Why sharding is great: demystifying the technical properties," 2021. Note: The blockchain trilemma (decentralization, security, scalability) was informally described by Buterin in multiple posts and talks beginning ~2017; no single canonical paper exists.

[37] Cambridge Centre for Alternative Finance, "Cambridge Bitcoin Electricity Consumption Index (CBECI)," University of Cambridge, 2025. Available: https://ccaf.io/cbnsi/cbeci

[38] Ethereum Foundation, "Ethereum Staking Statistics," beaconcha.in, 2025. Available: https://beaconcha.in/

[39] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," in Advances in Cryptology - CRYPTO '87, Springer, 1987, pp. 369-378.

[40] Mina Foundation, "Mina Protocol: A Succinct Blockchain," 2020. Available: https://minaprotocol.com/

[41] S. Micali, M. Rabin, and S. Vadhan, "Verifiable Random Functions," in Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 1999, pp. 120-130.

[42] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," Journal of Cryptographic Engineering, vol. 2, no. 2, pp. 77-89, 2012.

[43] R. Dahlberg, T. Pulls, and R. Peeters, "Efficient Sparse Merkle Trees: Caching Strategies and Secure (Non-)Membership Proofs," in Proceedings of the 21st Nordic Conference on Secure IT Systems (NordSec), Springer, 2016.

[44] B. Kilic, "Rate-Limiting Nullifier (RLN)," Ethereum Foundation Privacy and Scaling Explorations (PSE), 2022. Available: https://rate-limiting-nullifier.github.io/rln-docs/

[45] C. Dwork, N. Lynch, and L. Stockmeyer, "Consensus in the Presence of Partial Synchrony," Journal of the ACM, vol. 35, no. 2, pp. 288-323, 1988.