ROKO Network: A Proof-of-Accurate-Time Settlement Layer for Verifiable Machine Work

Conner Chevaillier · Alexander Roberts[1] · Joseph Magly[2] · Stalker

[1]Alexander@roko.network [2]jmagly@roko.network

ROKO Network

Important Notice: *This paper is a living document that will be actively updated as ROKO Network evolves. We welcome contributions from the open source community to expand and refine its content.*

Abstract. ROKO Network presents a layer-1 blockchain that treats time as something the network *measures and proves* rather than a number the block producer suggests. Conceived for robots and industry first — precision time, synchronization, and attestation for physical systems — the network now also serves the fastest-growing workload to join that industry: autonomous agents performing real, billable work. Machine work at scale — whether the machine is a robot on a factory floor, a sensor on a pipeline, or an agent in a datacenter — requires three guarantees that no single vendor can credibly self-certify: independent, nanosecond-resolution time; tamper-evident, replay-protected sign-off that a specific unit of work happened exactly once; and neutral settlement that ties payment to verified completion. ROKO is built on the Polkadot SDK with full Frontier EVM compatibility, securing block production and finality with BABE and GRANDPA while Proof of Accurate Time (PoAT) operates as a consensus modifier: validators run a peer-to-peer time mesh, per-validator time quality is recorded on-chain, every transaction receives a signed temporal receipt with a nanosecond canonical timestamp, and deterministic fee-priority ordering with consensus-enforced inclusion deadlines removes block-producer discretion over ordering and makes censorship provable. Around this core, ROKO integrates a standards-based traceable-execution stack — an Agent2Agent (A2A) execution substrate, a W3C PROV-DM packaging layer with content addressing, and CustodyCore, the ROKO organization's FROST-MPC custody and anchoring bridge — plus a unified metering framework in which every lifecycle event on packaged data becomes a stampable, meterable, billable event. Token demand is therefore workload-driven: it scales with executions anchored, time-units metered, and transactions settled. Governance follows the tradition of self-amending, on-chain-governed networks: chain emissions in the native coin reward governance token holders and fund the network treasury, while the native coin pays for operations. Collectively, ROKO Network aims to be the neutral clock, notary, and clearing layer for the machine economy — from factory floors and vehicle fleets to the autonomous agent economy.

1 Introduction

Machines have done billable work for decades. Robots assemble, sensors monitor, industrial controllers run production lines — and industry learned long ago that coordinating machines demands precise, shared time. What is new is the workforce: autonomous software agents are moving from demonstration to deployment, writing code, processing documents, transforming media, operating infrastructure, and transacting with other agents — and an increasing share of that work is *billable*: performed by one party's machines on behalf of another, under commercial terms. This expansion of the machine workforce exposes a structural gap. When a human performs work, an employment relationship, an invoice, and the legal system supply accountability. When a machine performs work — often in seconds, often thousands of times per day, often for a counterparty it has never interacted with before — there is no equivalent substrate for proving *what* was done, *when* it was done, and that it was done *exactly once*, in a way both parties can trust without trusting each other.

Today, that gap is filled by self-certification: the vendor's logs, the vendor's clock, the vendor's word. Self-certification does not survive adversarial conditions, and it does not scale to an economy in which agents from different organizations transact autonomously. What is required is a neutral third position — an authority nobody controls unilaterally — that can attest to time, to occurrence, and to settlement.

This paper presents ROKO Network, a layer-1 blockchain designed to occupy exactly that position. ROKO's genesis is industrial: the network was conceived for robots and industry first — a public source of precision time, synchronization, and attestation for physical systems — and autonomous agents have simply become part of the industry it was built to serve (Section 9.3). The ambition is deliberately democratic: capabilities that today exist almost exclusively inside large industrial companies — precision synchronization, attested telemetry, machine-scale settlement — offered as open, public infrastructure accessible to the smallest robotics and AI teams. ROKO is an EVM-compatible chain built on the Polkadot SDK with the Frontier Ethereum layer, so existing Solidity contracts and Ethereum tooling work unmodified — and on that familiar base, the network adds what no conventional chain provides: a validator time mesh that measures and agrees on time, signed temporal receipts on every transaction, deterministic ordering with consensus-enforced inclusion, and a consensus time oracle readable from contracts and RPC without a third party

to trust. We present ROKO not as an insular platform, but as the settlement backbone for an open, standards-based traceable-execution stack that can operate and grow via the open-source community: the execution contract is the Agent2Agent (A2A) standard, frozen and conformance-tested; the provenance model is W3C PROV-DM; receipts are independently re-verifiable by any third party. This study presents an in-depth exploration of ROKO Network's core technologies, its economic model, and its potential to make blockchain settlement the default clearing mechanism for machine work at scale.

2 Background & Related Works

Bitcoin [1] and Ethereum [2], as pioneering blockchain networks, established the two primitives this work builds on: a neutral ledger that no single party controls, and programmable settlement over that ledger. Subsequent protocols have extended throughput, programmability, and interoperability. However, a holistic solution for *verifiable machine work* — precise attestation of when work occurred, proof that a specific work-artifact existed at that moment, and settlement bound to that proof — has remained elusive. This gap is addressed by ROKO Network.

In our analysis, the quintessential hurdles impeding a machine-work economy encompass (1) **self-certification**, the inability of any single vendor to credibly attest to its own work; (2) **temporal imprecision**, the coarseness and manipulability of existing timestamp sources relative to the demands of metered, ordered, high-frequency machine work; (3) **provenance opacity**, the absence of standardized, machine-readable records of who or what produced an artifact and from what sources; and (4) **settlement friction**, the lack of a neutral mechanism tying payment to verified completion between mutually distrusting counterparties.

2.1 The Self-Certification Problem In any two-party exchange of machine work, the seller's telemetry is the default record: their logs say the job ran, their clock says when, their database says the output is authentic. Each of these claims is unauditable by construction — the party producing the evidence is the party whose performance the evidence attests. Haber and Stornetta identified the core requirement as early as 1991: a digital timestamp is only meaningful if it is infeasible for *anyone*, including the timestamper, to back-date or forward-date the document [3]. Centralized time-stamping authorities (RFC 3161 [4]) partially address this but reintroduce a single trusted party and offer no settlement path. Public blockchains distribute the trust but, as discussed below, provide time that is too coarse and too producer-influenced for metering machine work.

2.1.1 Timestamping and Ordering in Existing Systems Network Time Protocol [5] synchronizes clocks but attests nothing: an NTP-synced server can still lie about when an event occurred. Lamport established that ordering in distributed systems is a first-class problem requiring explicit mechanism rather than trusted clocks [6]. Blockchain block timestamps inherit both limitations — they are set by block producers within tolerance windows measured in seconds, are manipulable at the margin by the producers themselves, and offer granularity orders of magnitude too coarse for metering work that completes in milliseconds. Oracle networks can import external time attestations but concentrate trust in the oracle operator set and add an additional failure surface. What is missing is a consensus mechanism in which *precise time itself is the attested product* — where the network's honest majority stands behind nanosecond-resolution statements of when an event was observed and finalized.

2.2 Metering and Precision Billable machine work is metered work. Industry learned this decades before agents did: power metering, production-line coordination, and trade-event recording all depend on synchronized clocks that both sides of a transaction accept. Charging per execution, per duration, or per access requires measurements both parties accept. Sub-second work makes second-granularity timestamps useless for duration metering; adversarial incentives make self-reported durations unacceptable. A neutral, high-precision time authority converts raw telemetry into *claims that can be settled*: this task began at T_0 and finalized at T_1, attested by an authority set nobody controls unilaterally. Precision is therefore not a vanity metric — it is the difference between a meter both sides can bill against and a log one side must simply believe.

2.3 Provenance and Interoperability Standards A settlement layer for machine work is only trustworthy if the work itself is described in open, verifiable formats. Two standards anchor the stack described in this paper. The W3C PROV data model [7] provides machine-readable provenance — `wasGeneratedBy`, `wasDerivedFrom`, `wasAttributedTo` — recording which agent, using which model, produced which artifact from which sources. The Agent2Agent (A2A) protocol [8] — contributed by Google to the Linux Foundation — standardizes the task lifecycle between agents, enabling signed, conformance-testable execution contracts. Building on published standards rather than proprietary formats means buyers and counterparties can verify the layer without trusting any single vendor — which is precisely the condition under which a neutral chain has a job to do.

2.4 Workload-Driven Token Economies Recent networks such as Bittensor [9] demonstrate a token model in which demand derives from selling a real machine workload — compute and machine-learning inference priced and settled in the network's token — rather than from transaction speculation alone. ROKO follows the same model: the network is designed as an abstraction layer that coordinates decentralized compute primitives [17], ultimately selling general compute and precision-time utility priced and settled in its native coin — and its DAO already operates in this economy today, deploying nodes across decentralized-infrastructure networks and running validators (including on Bittensor and Ethereum) whose rewards fund the network treasury. Traceable agentic execution is the lead workload — the fastest-growing category of machine work, and the one whose verification demands (time, occurrence, settlement) map most directly onto what the network provides — but the same anchoring, metering, and settlement rails price any machine workload. Section 7 details the economic model; Section 9 the product line.

3 ROKO's Layer 1: Consensus and the Time Mesh

3.1 Background & Issues with Blockchain Time Conventional consensus mechanisms treat time as an input — a loosely validated field the block producer supplies — rather than as an output the network attests. Producer-set timestamps are acceptable when time merely orders blocks; they are insufficient when time is the *product*: when counterparties need to know, to high precision and with cryptographic accountability, when a specific transaction or artifact was observed by the network. A network whose purpose is metering and ordering paid machine work must make temporal accuracy a first-class, consensus-visible property. Industries that live on precise time — production lines, power grids, trading venues — solved *distribution* long ago with PTP and disciplined GNSS references; what those private time fabrics cannot provide is *attestation*: proof to an outside counterparty. Blockchains have the inverse problem — public attestation without precision. ROKO's temporal layer fuses the two.

3.2 Foundation: BABE and GRANDPA ROKO does not replace consensus with something exotic. The chain is built on the Polkadot SDK [13]: block production is BABE (slot-based, with primary and secondary slots) and finality is GRANDPA [14] — the same battle-tested machinery securing Polkadot — with a 3-second block time on the mainnet runtime. Validator selection operates under Nominated Proof-of-Stake with multi-phase election, and the session key set carries a ROKO-specific temporal key alongside the standard BABE/GRANDPA identities, so a validator's time attestations are bound to the same identity that produces and finalizes blocks.

3.3 Proof of Accurate Time: Time as a Consensus Input Proof of Accurate Time (PoAT) is layered onto this foundation as a **consensus modifier** — a pipeline from physical clocks to on-chain consequences. First, *the time mesh measures*: validators run PTP Squared, a native peer-to-peer time-synchronization protocol descended from the IEEE 1588 Precision Time Protocol lineage [15], continuously probing each other's clocks over the network, estimating offsets from the lowest-latency exchanges, scoring peer reputations statistically (so the mesh distrusts bad clocks rather than merely averaging them), and converging on a single **mesh consensus time**. Second, *an inherent puts it on-chain*: each block carries a time-mesh snapshot consumed by the runtime, which stores per-block and per-validator time quality as a fixed-point score and records periodic health checkpoints. Third, *quality feeds back into consensus*: PoAT is designed so that measured time quality influences block-production eligibility and rewards — making accurate time a condition of full participation rather than a courtesy. Time-quality offences (persistent drift, contradictory offsets, low reputation) are defined with slash fractions and integrated with the standard offences machinery; enforcement is being enabled in stages as the mesh matures (Section 11.5).

3.4 Validator Time Sources and Tiers Validators do not all need laboratory-grade hardware — they need to be honest about what they have. Each validator self-classifies its time source, reporting a measured root distance to UTC in nanoseconds: **Anchor** tier (sub-microsecond root distance; GNSS receiver with PPS-disciplined clock), **Standard** tier (NTP-disciplined, sub-10-microsecond), and **Minimal** tier (plain system clock). Classification is automatic — the node inspects the host's time daemon, PPS devices, and NIC hardware-timestamping capability — and a production network requires only a small number of Anchor validators for the time consensus to be Byzantine-fault tolerant. The tier model descends from the network's **Proof-of-Capability** design [20]: stratum-based accuracy thresholds in which nodes are scored on time accuracy, uptime, and integrity, rewarded in proportion to the capability they demonstrably maintain, and demoted when they drift outside their declared class. The hardware floor is deliberately modest: Anchor-tier timing hardware costs tens to hundreds of dollars, and a Raspberry Pi–class reference validator with a GNSS module runs on the network's testnet today.

3.5 Temporal Transactions: Receipts, Deadlines, and Deterministic Ordering On ROKO, every transaction — Substrate extrinsic or Ethereum transaction — acquires a cryptographic paper trail, with no wallet changes and no new transaction type. Three mechanisms combine. *Temporal receipts:* when a transaction is admitted to the pool, the node issues an ECDSA-signed temporal receipt binding the transaction to a nanosecond admission time, signed under the validator's temporal key. *The inclusion deadline:* each receipted transaction carries a deadline (15 seconds by default), checked at block import — if a block omits a receipted transaction whose deadline has passed, honest validators reject the block entirely. This is an inclusion-enforcement bound, not a latency promise: a block producer cannot silently censor a receipted transaction, because past the deadline, leaving it out invalidates the block. *Deterministic fee-priority ordering:* canonical timestamps are assigned by a protocol-level queue that stamps pending transactions in descending fee order on a fixed tick — higher fees earn earlier canonical timestamps under a published rule, with per-block temporal ordering enforced at finalization against a monotonic chain-wide watermark. Timestamps throughout are **NanoMoments** — unsigned 128-bit nanosecond counts since the Unix epoch. One distinction the network states explicitly: nanosecond figures are *resolution*, not an accuracy claim for any individual clock — accuracy is separately measured, and every consensus-time reading is published together with the network's own time-quality score and convergence state.

3.6 Anchoring Receipts and Independent Verification For external workloads that settle on ROKO (Section 8), the chain's temporal machinery is consumed through a second, settlement-grade receipt surface: the **anchoring receipt**. A content hash submitted to ROKO is returned as a signed envelope binding the hash to network time — carrying the chain and genesis identifiers, the authority set and epoch that attested it, the block height and extrinsic hash of the anchoring transaction, nanosecond observed and finalized times, and a convergence status. The receipt is self-contained: any third party holding the receipt and the artifact can re-derive the content hash and verify the signature against the published authority set — offline, without querying ROKO, and without trusting the party that presented it. The reference verifier (implemented in CustodyCore, Section 8.1.3) independently re-checks payload hash match, replay uniqueness, chain and author-

ity binding, time quality, and convergence, with an explicit failure taxonomy. Replay protection guarantees the property settlement depends on: *this exact* unit of work is signed off *once*.

3.7 Security Model and Misbehavior The network's guarantees rest on explicit trust assumptions rather than optimism: block production and finality assume an honest majority of bonded validators; the inclusion-enforcement guarantee of Section 3.5 holds under an honest majority at block import; and the time consensus is Byzantine-fault tolerant, requiring only a small set of honest Anchor-tier validators (Section 3.4). Within those bounds, each class of misbehavior has a defined detection path and consequence.

Consensus equivocation — producing conflicting blocks or finality votes — is handled by the standard offences-and-staking machinery the Polkadot SDK runtime ships: offences are reported on-chain and slash bonded stake, with slashed backing routed to the Treasury (Section 7.2).

Temporal dishonesty — persistent drift, contradictory offsets, low mesh reputation — is detected twice: statistically inside the mesh, whose reputation weighting discounts bad clocks before they can influence consensus time, and on-chain, where per-validator time quality is recorded every block. Time-quality offences are defined with slash fractions and integrated with the same offences machinery; as Sections 3.3 and 11.5 state plainly, their enforcement is deliberately staged and currently disabled while mesh parameters are validated under load — the operative deterrent today is reputational (recorded quality, tier demotion) rather than economic.

Censorship and reordering are resolved structurally rather than punitively: ordering is fixed by a published rule before block assembly, and a block that omits a receipted transaction past its deadline is rejected outright by honest validators (Sections 3.5, 4.1) — the misbehavior produces an invalid block, not a profit.

Replay and double-billing are closed by replay protection in receipt verification and durable idempotency in the execution substrate (Sections 3.6, 8.3): the same unit of work cannot be signed off or settled twice.

Capability misrepresentation — a validator overstating its time source — is bounded by automatic classification: the node measures its own root distance and inspects its own hardware, and validators drifting outside their declared class are demoted (Section 3.4).

Sybil pressure is priced by Nominated Proof-of-Stake: influence over block production, finality, and the temporal key requires bonded stake, making identities expensive rather than free.

The residual trust item is governance-level rather than consensus-level: the gated-testnet sudo key disclosed in Section 5 remains the dominant consideration until progressive decentralization completes (Section 11.6).

4 The Temporal EVM Platform

ROKO's developer surface is deliberately unexotic: an EVM-compatible chain where MetaMask, Hardhat, ethers.js, and existing Solidity contracts work unmodified. The chain uses Ethereum-native 20-byte accounts at every level — there is no address translation between the Substrate and EVM layers — with EIP-1559-style fees and the standard Ethereum JSON-RPC surface served on a single port. EVM `block.timestamp` remains the standard seconds-level value, so existing contracts behave identically; nanosecond temporal data is strictly additive, exposed through dedicated surfaces.

Those surfaces are what make the platform temporal. A `temporal_*` JSON-RPC namespace exposes the network's clock and ordering state — consensus time with quality and convergence metrics, per-transaction canonical timestamps (queryable by either a Substrate extrinsic hash or an Ethereum transaction hash, so EVM tooling can look up timestamps by the hash it already has), per-block temporal metadata, queue statistics, and watermark state. On-chain, contracts read the same state through custom precompiles: a **Temporal precompile** exposing a Solidity `ITemporal` interface (`getConsensusTime()`, `getTransactionTimestamp(bytes32)`, `getWatermark()`, and related calls returning `uint128` NanoMoments), and a **pwROKO precompile** exposing the staking token (Section 7.2) through a standard ERC-20-style interface. The result is a consensus-backed time oracle with no third party to trust — readable from Solidity, from any Ethereum client library, and from ordinary `curl`.

4.1 Ordering Manipulation and Censorship Resistance The temporal machinery of Section 3.5 yields a concrete answer to transaction-ordering manipulation — not a resistance slogan but two verifiable mechanisms. First, ordering is decided by a *deterministic, transparent rule applied at receipt* — descending fee order under a protocol-level queue — rather than by validator preference or a private builder auction. A block producer cannot quietly reorder, front-run, or sandwich transactions it has already received, because the order was fixed and sealed in signed receipts before block assembly. Second, inclusion is *enforced*: block import rejects any block that omits a receipted transaction past its deadline, so censorship stops being deniable — either the transaction is included, or honest validators reject the block that excluded it. The network is precise about scope: fee priority still exists, by design — what is removed is producer *discretion*; the mempool is not encrypted; and the guarantees hold under an honest majority at block import. These properties are auditable by anyone over the public RPC surface.

5 Governance Structure

ROKO Network's governance today operates through the **ROKO DAO**: governance token holders coordinate decision-making on product development and resource allocation through token-weighted proposal voting, with the DAO treasury held in publicly verifiable multisig accounts. The governance token is **$ROKO** — held by the network's current holders — and the token was launched without any raise, ICO, or token sale. Commissioned governance research [16] maps the evolution path toward a **constitution-based, multicameral model** designed to resist plutocratic influence and reward quality contribution: a DAO Constitution codifying membership, decision-making, treasury allocation, and

dispute resolution; a **multi-token governance system** in which the non-transferable staking token (pwROKO, Section 7.2) carries voting rights, soulbound reputation tokens capture the quality and history of contribution, and donor-recognition tokens acknowledge financial support; specialized **working groups** with delegated authority over their domains; and role-based proposal classification — system-level proposals under weighted voting, community proposals under simple majority — with voting mechanics evolving from token-weighted toward quadratic and reputation-based schemes as the community scales. Treasury management follows a commissioned strategy [18] built on three pillars — sustainability, growth, and security — with tiered short/medium/long-term liquidity management and funding drawn from staking fees, validator rewards, marketplace fees, and the DAO's own node and validator operations.

On-chain, the network's runtime already ships a full governance pallet surface — council and technical-committee collectives, democracy and referenda, conviction voting, treasury, bounties, and tips — in the tradition of self-amending chains such as Tezos and the governance systems of Substrate-based networks: protocol changes proposed, voted on, and enacted on-chain by governance token holders. Governance participation is rewarded through chain emissions: the network emits its **native coin**, with a portion of each emission routed to the on-chain **Treasury** and a portion distributed to governance token holders.

The native coin is the network's operational currency: it pays for running agentic processes and machine workloads on the network and for network and time-network fees (Section 7). This emissions-and-fee model is not yet live on-chain; it is planned for a forthcoming network update, and its parameters — the emission schedule, the treasury/holder split, and the proposal and voting mechanics — will be specified in a subsequent revision of this living document. During the current gated-testnet phase a sudo (root) key remains active in the runtime — a standard posture for a network at this stage, disclosed plainly — and authority sets for the time network rotate in epochs, with each epoch's set and keys published on-chain so that every temporal receipt remains verifiable against the authority configuration that produced it, including historically, after rotation.

6 Scaling and Network Expansion

ROKO's scaling posture follows its demand profile. The dominant transaction classes — anchoring submissions and settlement transactions — are small, uniform, and batch-friendly, which shapes the scaling design in three ways.

6.1 Demand-Proportional Throughput Anchoring demand scales with the number of traceable work-units produced by the connected ecosystem. Because each anchor is a fixed-size content hash, anchoring transactions are small and uniform, and substantial workload growth is absorbed within existing block capacity. Growth beyond that envelope is addressed by batching before it is addressed by protocol change.

6.2 Aggregation and Batching Because an anchor is a fixed-size content hash, high-volume producers can aggregate: many work-artifacts can be combined into a single anchored manifest whose per-item checksums live off-chain in the packaging layer (Section 8.1.2), preserving per-item verifiability while consuming one on-chain anchor. The Knowledge Shard format — a checksummed, self-contained bundle with a manifest of per-section hashes — is the natural aggregation unit, and the phased on-chain data ladder (Section 11.1) extends this from anchored manifests to published ones. Batch settlement follows the same pattern: settlement transactions referencing verified receipts can clear in batches where counterparties permit.

6.3 The Time Network as the Scaling Invariant Whatever the transaction volume, the property that must not degrade is time quality. Routing a dedicated share of all fee revenue to the TemporalFeePool (Section 7.3) ties network revenue growth directly to timestamper funding, so that scale improves rather than erodes the precision the network sells.

7 The ROKO Economy: Coin, Tokens & Fees

7.1 Coin Utility and Fee Surfaces The ROKO native coin is the asset in which all network services are priced and settled: it pays for operations — running an agentic process or machine workload on the network — and for network and time-network fees. (Today's $ROKO token carries governance rights and receives a share of native-coin emissions, per Section 5; native-coin fee payment activates with a forthcoming network update.) Five value-capture mechanisms consume the coin, ordered by how directly each is wired today:

1.) Anchoring / sign-off fees: Every traceable unit of work that receives a temporal receipt pays a fee in the native coin. High-volume agentic and machine execution produces recurring, usage-based anchoring demand.

2.) Settlement fees: Completed, verified work clears on-chain — direct pay-on-verified-completion transactions from day one (Section 8.2) — with a fee per settled unit of work.

3.) Time-network metering: Precision timing sold as a metered service, per-event or per-duration, settled on-chain (Section 11.3).

4.) Staking / network security: The validator set securing the time-and-settlement layer earns yield from securing real economic activity rather than speculative volume alone.

5.) Public attestation services: Paid, publicly auditable attestations — identity, conformance, work-completion — offered as network services (Section 11.4).

The demand thesis for token holders follows directly: consumption scales with executions anchored, time-units metered, and transactions settled — that is, with adoption of the integrated stack (Section 8) and the growth of agentic work itself. The chain is monetized by being used as designed.

7.2 The Token System: $ROKO, Native ROKO, and pwROKO Three assets play distinct roles. **$ROKO**, the existing fixed-supply token held by the community (369,369,369,369 total supply, with the Foundation's allocation managed by the DAO in multisig custody), is the governance asset: it votes in the

DAO today and receives a share of native-coin emissions under the planned model (Section 5). The **native ROKO coin** (18 decimals, Ethereum-style accounts) is the chain's operational currency — gas, fees, anchoring, and settlement. **pwROKO** ("wrapped ROKO") is the staking token: locking 1:1 mints pwROKO, which is deliberately **non-transferable** — it can only be minted by locking and burned by unlocking — so that staking power cannot be traded as a liquid derivative, while delegation to validators is handled as an internal assignment that never transfers the token [17]. The commissioned token-economy design [17] assigns pwROKO a second role beyond staking: it is the intended **voting token** of the multi-token governance model (Section 5), concentrating governance rights in committed, staked holders, and the intended access key for premium product features — a Web3 alternative to freemium tiers. Staking is denominated in pwROKO in the live runtimes; unlocking is a deliberate two-step flow with a governance-adjustable cooldown (14 days at mainnet parameters, matching the design study's recommendation), and slashed stake routes its native backing to the Treasury. Reward emissions are designed to be paid in pwROKO and drawn from a pre-allocated, treasury-locked pool on a front-loaded, multi-year declining schedule — keeping fully-diluted supply constant and outflows predictable [17]; the live runtimes implement a standard staking-inflation curve (bounded between 2.5% and 10%, targeting an ideal staking ratio) as runtime parameters under governance, not an APY promise. There is currently no bridge between the existing $ROKO token and the chain's native coin; the mapping between them is part of the planned network update described in Section 5.

7.3 Fee Routing and the TemporalFeePool Every transaction fee splits three ways at the runtime level, as illustrated in Figure 1: 50% to the **block author**, rewarding block production and liveness; 30% to the **TemporalFeePool**, which is periodically distributed to timestamping validators in proportion to their recorded timestamping contribution; and 20% to the on-chain **Treasury**, funding ecosystem development under governance. The TemporalFeePool is the mechanism that converts network usage into time-network quality: where a general-purpose chain funds only computation, ROKO's fee routing pays the validators whose disciplined clocks and attestations *are* the product. This creates a flywheel — usage funds time quality, time quality earns anchoring trust, anchoring trust drives usage — and gives holders a legible answer to why fees exist at all: they are the purchase price of neutral time, paid to the parties who produce it. Two fee-policy levers are planned as anchoring workloads scale: a discounted fee class for anchoring transactions (the volume workload), and an increased TemporalFeePool share — both governance-adjustable, and both aimed at the same goal of maximizing the adoption flywheel while keeping the time network funded. The commissioned token-economy design [17] adds a complementary set of demand-side levers under governance control — product revenues collected in fiat or ROKO with market buybacks, burn mechanisms at the staking layer, and early-unstake fees routed to the treasury — forming a sustainability loop in which product revenue, staking, and buy pressure reinforce one another.

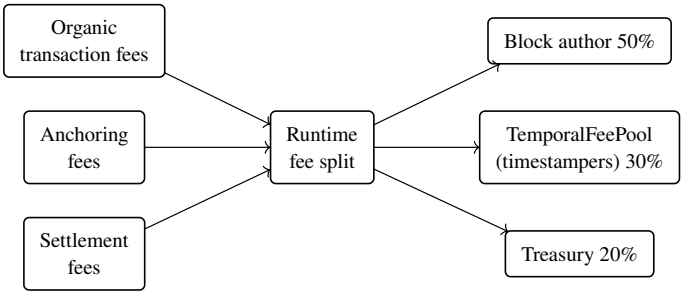

Figure 1: Diagram of ROKO's fee-distribution system.

7.4 Participant Rewards at a Glance The reward mechanics above are distributed across the consensus, governance, and economic sections; Table 1 consolidates them into a single map of who earns, from which flow, on what basis, and what is live

Participant	Reward source	Basis	Status
Block-producing validators	50% of every transaction fee, plus staking rewards	Fee share per block authored; staking-reward inflation curve bounded between 2.5% and 10%, targeting an ideal staking ratio — a runtime parameter under governance, not an APY promise (§7.2)	In runtime today (gated testnet)
Timestamping validators	30% of every fee via the TemporalFeePool	Distributed periodically in proportion to recorded timestamping contribution; higher-tier time sources (Anchor) sustain fuller participation (§3.4, §7.3)	Fee split in runtime; consensus-consequence feedback staged (§11.5)
pwROKO stakers and delegators	Staking rewards, designed to be paid in pwROKO	Proportional to stake and delegation; drawn from a pre-allocated, treasury-locked pool on a front-loaded declining schedule, keeping fully-diluted supply constant (§7.2) [17]	Staking live in both runtimes; emission-pool design proposed [17]
$ROKO governance holders	Share of native-coin chain emissions	A portion of each emission to holders, a portion to the Treasury; split and schedule to be specified with the activating network update (§5)	Planned — future network update
DAO / Treasury	20% of every fee + emission share + the DAO's own node and validator operations	Funds ecosystem development under governance; managed per the commissioned treasury strategy (§5) [18]	Fee routing in runtime; emission share planned
Compute and capacity providers	Payment for workloads sold through the product suite	Proportional payment with verifiable task completion (Compute Orchestrator); marketplace monetization of datasets, models, and agents (§9.1) [19]	Design / roadmap

Table 1: Participant reward map: sources, bases, and implementation status.

today. It is a map of *sources*, not a forecast of *magnitudes* — the distinction is drawn explicitly below the table.

What Table 1 deliberately does not contain is magnitudes. The native-coin emission rate and the treasury/holder split are unspecified until the network update that activates them (Section 5); fee prices for the five surfaces of Section 7.1 are not yet set; and the TemporalFeePool's distribution period and contribution metric are runtime parameters to be published with the mainnet configuration. Where the commissioned token-economy design proposes concrete shapes — an emission pool paid in pwROKO on a front-loaded schedule declining over roughly an eight-year span, with incentive parameters reviewed every three to six months [17] — these are design recommendations under governance, not commitments. Prospective participants should read this section as the complete list of reward faucets and their funding sources; per-participant economics become computable when the emission schedule, fee prices, and distribution parameters are published.

8 The Traceable Work Settlement Layer

ROKO Network anchors the world's first end-to-end **Traceable Work Settlement Layer** (TWSL) — an open, standards-based stack in which any unit of machine work can be executed under a signed contract, packaged with verifiable provenance, anchored to neutral time, and settled on-chain. The TWSL is made possible by three components: a conformance-tested execution substrate, a universal data-packaging layer, and CustodyCore — the ROKO organization's custody and anchoring bridge — with ROKO's layer-1 serving as the time and settlement authority beneath all three (Figure 2).

The components interoperate by design while remaining independently owned and separately governed: the execution and packaging layers are complementary open-ecosystem infrastructure, and CustodyCore is ROKO-organization software (AGPL-3.0). This is a deliberate architecture — *complementary by design* — rather than a merged product: each layer is independently useful, independently verifiable, and connected through open seams any implementer can adopt.

The stack also arrives with a working ecosystem behind it. The execution substrate is part of the **Agentic Operating System** (AOS) — a fully open-source portfolio of agentic infrastructure, including the AIWG agentic-workflow framework, that teams adopt on its own merits, with no commercial relationship required. Every AOS/AIWG deployment that opts into traceable execution becomes a workload source for ROKO: its agent runs produce the anchoring, metering, and settlement transactions of Section 7.1, and its data operations flow through the packaging layer's meterable lifecycle events (Section 10). The utilization thesis is therefore ecosystem-shaped rather than customer-shaped — coin demand and network revenue scale with the breadth of integrated-stack adoption across the open agentic ecosystem, not with any single account.

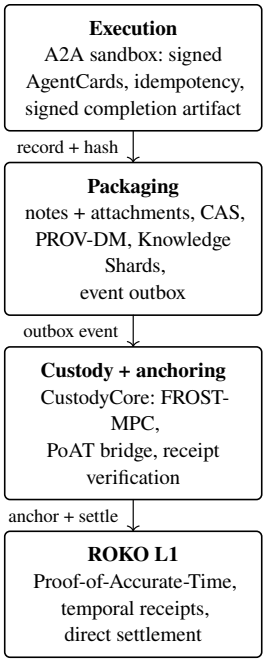

Figure 2: The four layers of the Traceable Work Settlement Layer.

8.1 The Components of the TWSL The following sections delve into the three major components that make up the TWSL and explain how they collectively facilitate verifiable, settled machine work on ROKO Network.

8.1.1 The Execution Substrate (A2A) Work enters the TWSL through execution environments implementing the Agent2Agent executor contract — the reference implementation being the open-source `agentic-sandbox`, whose contract is frozen and verified by a public conformance harness any third party can run for a neutral pass/fail attestation. Every task is bound to a specific runtime instance; instances publish Ed25519/JWS-signed AgentCards; execution images are pinned by SHA-256 digest; and a durable idempotency mechanism provides anti-double-execution guarantees — the off-chain half of the exactly-once property that ROKO's replay protection completes on-chain. Millisecond-timestamped lifecycle events give each unit of work a measurable execution span, and on completion the runtime emits a signed completion artifact — task identifier, runtime instance, result hash, exit code, completion time, signed by the instance's AgentCard key — binding payment to a specific, verifiable execution.

8.1.2 The Packaging Layer (Fortemi) Between execution and chain sits a universal data-packaging layer: Fortemi, a production knowledge-management system (50 tagged releases) that functions as the stack's normalizable envelope format for arbitrary data. Anything — text, code, media, documents, model outputs, sensor streams, machine telemetry — is packaged as a note with attachments in a BLAKE3 content-addressed blob store; every version carries a SHA-256 content hash; provenance is recorded as W3C PROV-DM by name (`wasGeneratedBy`, `wasDerivedFrom`, `wasAttributedTo`), capturing which agent, using which model, produced what, when, from which sources. Portable **Knowledge**

Shards — checksummed, manifest-carrying bundles with per-section hashes and a versioned migration engine — provide the aggregation and transport unit. Identity uses wallet-style Base58Check addresses; multi-recipient envelope encryption (X25519 key agreement, AES-256-GCM payloads [10][11]) lets private data be chain-referenced without being chain-readable. An atomic transactional event outbox emits every lifecycle event exactly as recorded — the integration seam the anchoring layer consumes. The consequence of this design is the TWSL's generality: the settlement flow below is not restricted to sandbox agent runs. If it touches the stack, it can be packaged; if it is packaged, it can be chain-tied.

8.1.3 Custody and Anchoring (CustodyCore) CustodyCore is the ROKO organization's custody, signing, and anchoring layer. It provides FROST threshold signing [12] (secp256k1/Taproot and Ed25519) over arbitrary content, with key shares that never leave participant devices — the coordinator sees only commitments and signature shares — plus a cryptographic trust graph and attestation model. Its Proof-of-Accurate-Time bridge implements the full anchoring pipeline: canonical bytes → Keccak-256 content hash → submission to ROKO → anchoring receipt → independent re-verification (Section 3.6), backed by a durable, crash-safe anchor worker with receipt persistence. CustodyCore is the component that makes the ROKO organization an active builder of its chain's utility layer rather than a passive oracle awaiting adoption.

8.2 Mechanisms of Settlement Settlement within the TWSL operates through a coordinated sequence designed so that no funds are held and no party is trusted: completion is proven, then payment clears.

1. A buyer submits a task to an execution environment under the A2A contract — signed AgentCard, idempotency key.
2. The agent executes in an isolated runtime; the work and its full provenance (agent, model, lineage, sources) are recorded in the packaging layer and content-hashed; the event outbox emits the completion atomically.
3. The runtime returns a signed completion artifact to the buyer, including the result hash.
4. The anchoring consumer forwards the entity identifier and content hash to CustodyCore, which submits the hash to ROKO's Proof-of-Accurate-Time and receives an anchoring receipt — nanosecond-resolution, authority-bound, replay-protected.
5. CustodyCore independently verifies the receipt: hash match, replay uniqueness, authority binding, time quality, convergence.
6. On a verified receipt, a **direct settlement transaction** clears on ROKO — receipt reference, payer, payee, amount, work reference — paying on verified completion.

Read as one sentence: an agent does work in a signed, traceable sandbox; the packaging layer records who/what/when/from-what and content-hashes it; CustodyCore anchors that hash on ROKO; ROKO returns a verifiable receipt; a direct transaction settles payment on verified completion. The day-one model is deliberately escrow-free — direct transactions minimize the settlement layer's build surface and its custody risk simultaneously. Escrowed and conditional exchange, for use cases that require held funds and staged release, is an additive Phase 2 capability (Section 11.2).

8.3 Key Innovations and Benefits Beyond binding payment to proof, the TWSL introduces several properties that distinguish it from existing approaches to work verification and settlement.

First, **evidence is independently verifiable end to end.** The execution contract is conformance-testable by third parties; provenance follows a W3C standard; the anchoring receipt verifies offline against a published authority set. At no point must a counterparty trust a vendor's logs, clock, or database.

Second, **the exactly-once property is enforced at two layers** — durable idempotency in the execution substrate and replay protection in receipt verification — closing the double-billing vector that self-certified systems leave open.

Third, **privacy and verifiability coexist.** Envelope encryption in the packaging layer means an artifact can be anchored, referenced, and settled on-chain while remaining readable only to its authorized recipients: chain-referenced without being chain-readable.

Fourth, **anchoring doubles as tamper-evidence.** Systems that would otherwise build internal hash-chained audit logs can instead anchor their records to ROKO, gaining external, neutral tamper-evidence — converting an internal engineering cost into network demand.

8.3.1 Advantages Over Existing Solutions Centralized time-stamping authorities (RFC 3161) provide a signature over a hash and a time, but reintroduce a single trusted operator, offer second-at-best precision guarantees, and provide no settlement path. Oracle-mediated attestation imports external claims on-chain but concentrates trust in the oracle set and adds a failure surface between the event and the ledger. Generic smart-contract escrow provides settlement but no verification — it can release funds on a signal, but cannot itself attest that work occurred, when, or exactly once. The TWSL's differentiation is vertical integration of the three guarantees: time, occurrence, and settlement are supplied by the same neutral network, with each layer's evidence consumable by the next and the whole chain of custody independently re-verifiable.

8.3.2 Criteria for Becoming a Settled Workload The TWSL is an open framework; any workload can adopt it. The criteria for a workload to settle on ROKO are (1) work-artifacts must be content-hashed (any collision-resistant hash the anchoring layer supports); (2) completion must be attested by a signed artifact binding the result hash to an identified runtime or producer; and (3) lifecycle events must be emitted through a reliable channel an anchoring consumer can subscribe to (the reference pattern being an atomic transactional outbox). Workloads meeting these criteria inherit the full stack — anchoring, verification, and settlement — without bespoke integration. The criteria are deliberately indifferent to what the machine is: a robotic work cell emitting signed, hashed telemetry qualifies exactly as an agent emitting a signed completion artifact does. Workloads lacking signed completion or reliable event emission require a more bespoke integration path.

9 Applications

The same primitives — canonical nanosecond timestamps, deterministic ordering, signed receipts with inclusion enforcement, and a consensus time oracle readable from contracts — unlock application classes well beyond agentic settlement. Everything below builds with unmodified Ethereum tooling.

9.1 The Network Product Line ROKO's commissioned product strategy [19] stages a first-party product suite on these rails, each product a demand source for the coin: an **AI Agents Marketplace** — a supply-driven hub for crowdsourced, co-owned AI where datasets, models, and agents are curated, traded, and monetized with the reputation, staking, and verification mechanics the chain provides; a **Compute Orchestrator** selling compute — orchestrating contributed and partner GPU capacity with workload tracking, proportional payment, and verifiable task completion; a **Robotic Simulation Platform** — a collaborative hub for simulation models and training data under a freemium model whose premium tiers are unlocked by staked pwROKO; and **timing services** — precision time sold as a service to applications that need synchronized, attestable operations. A unifying design goal from the network's economic R&D is **single-currency billing through account abstraction**: users pay in one currency while the network's controller quotes, prices, and settles the underlying compute primitives on their behalf [17]. The staged rollout deliberately leads with the marketplace and orchestrator — the general-compute demand engines — with the settlement rails of Section 8 underneath all of them.

9.2 Application Classes Beyond the first-party line, the temporal primitives serve any builder:

Time-sensitive DeFi. On a conventional chain, the block producer decides transaction order — for auctions, liquidations, and order books, that discretion is the attack surface. On ROKO, ordering is fixed by a deterministic rule at receipt: batch auctions settle strictly by canonical timestamp, on-chain order books resolve sequence disputes by querying the network, and liquidation engines carry ordering rules that are auditable rather than asserted.

Fair launches and mints. Allocation follows a published, protocol-level rule with no private path around it, and signed receipts make exclusion provable — a mint cannot be quietly reordered or censored.

Verifiable timestamping and attestation. "This existed at time T" without a trusted party: anchor a hash, retrieve its canonical timestamp later — document-existence proofs, IP priority claims, and sensor or event attestations queryable by anyone.

Audit trails for regulated workflows. Consensus-enforced temporal records — each transaction's canonical timestamp, signed receipt, and per-block metadata — support trade-event recorders, custody handoff logs, and approval-chain registries with a reconstructible, tamper-evident timeline. (ROKO provides timestamps and ordering proofs; mapping records to a regulatory regime's requirements remains the application's job.)

Time-locked and scheduled contract logic. Vesting schedules, option expiries, and deadline-driven settlement keyed off consensus time — with quality and convergence metrics attached — instead of block heights or producer-set timestamps.

Agentic work settlement — the flagship workload, served end-to-end by the Traceable Work Settlement Layer of Section 8.

9.3 Cyber-Physical Control and Synchronization This is where the network began. ROKO's genesis is a public network conceived for robots and industry first — delivering the precision time, synchronization, and attestation that physical systems require — and autonomous agents have simply become part of the industry it was built to serve. The primitives this paper has applied to agent work were built for machines that move: a shared, attested clock; deterministic, tamper-evident event ordering; signed receipts that make command histories provable; and settlement rails for machine work regardless of whether the machine is a process on a server or an actuator on a factory floor. An engineering boundary is stated plainly: hard-real-time control loops run at the edge, on local deterministic hardware — the chain is not in the servo loop. What the network contributes is the layer around the loop: sub-microsecond time distribution to the edge over the IEEE 1588/OCP-TAP lineage the industry already speaks, and consensus-grade attestation of the commands issued, the telemetry produced, and the order in which both occurred.

For **industrial controls**, this is deliberately additive rather than rip-and-replace: plants already running PTP-based instrumentation can draw attested time from the mesh and anchor their records to it — quality-assurance monitoring, equipment-failure detection, and pipeline monitoring gain a neutral, tamper-evident timeline, and distributed manufacturing gains "proof-of-moment" attestation for sensor and camera records that runs on-premise, with only hashes leaving the building. For **fleet and multi-robot coordination**, a trust-minimized shared clock and verifiable event log replace the single vendor-cloud orchestrator as the coordination substrate: robots from different operators can sequence against consensus time, disputes over what was commanded and when become queryable rather than arguable, and autonomous actions can be gated on attested authorization — a provable chain of command for machines. Fleets operating where satellite navigation timing is degraded or denied can draw resilient time from the mesh itself. The design carries operational lineage: the team behind it has shipped SIL3-certified (IEC 61508) functional safety for fully-autonomous warehouse-robotics fleets operating alongside human workers, and coordination networks for the power industry — and the independent-monitor, bounded-autonomy, fail-safe patterns of certified machine safety inform how ROKO separates attestation from actuation. Making these capabilities public is the point: sub-microsecond synchronized time, certified-safety coordination patterns, and attested machine telemetry exist today almost exclusively inside large industrial companies, and ROKO packages them as open infrastructure with a deliberately modest hardware floor (Section 3.4) — putting the same instruments in the hands of the smallest robotics and AI startups.

The program ships reference hardware so this is buildable rather than aspirational [21]: **EGG** — the Experimental Generalized Gateway, a 275 TOPS edge compute node with multimodal sensor inputs — gives robotics teams a working reference plat-

form that consumes mesh time and produces attested telemetry; **DROPBEAR**, an open-source six-foot biped robot developed with **Hyperspawn Robotics** and **Point Blank**, is the open hardware testbed; and a real-time edge-controls architecture — deterministic control at the edge on the precision-time and machine-to-machine trust foundation — is in active development with Hyperspawn. The Robotic Simulation Platform (Section 9.1) closes the loop from simulation to hardware: models trained and validated in simulation carry attested provenance onto physical machines.

10 The Unified Metering and Billing Framework

ROKO Network employs a novel approach to usage economics that generalizes settlement from *completions* to *lifecycles*. With a universal packaging standard in place, every lifecycle event on packaged data becomes a stampable, meterable, billable event — one framework rather than per-feature billing.

10.1 Billable Event Classes The framework meters three event classes, each drawing on signals the packaging layer already records. **Creation:** job durations (estimated and actual), cost tiers, inference token usage, and artifact-creation events — including sensor captures and machine telemetry — become the claim "this artifact was produced at time T (nanosecond) with W measured work," anchored with the artifact itself. **Utilization:** per-artifact access logs — timestamped and attributed per source — along with access counts and retrieval or inference reads become metered access claims, per-event or per-duration, aggregated into settleable usage. **Sharing and distribution:** encryption-recipient grants, shard exports and imports, and cross-instance synchronization each become a stamped event — a provable chain of who received what, when — billable per grant.

10.2 The Metering Pipeline Lifecycle events flow from the packaging layer's atomic event outbox into a usage meter, are stamped as PoAT-attested usage claims, aggregated, and settled on ROKO. The nanosecond time network is what makes the meter *neutral*: when something happened, and how much work went into it, is attested by an authority set nobody controls unilaterally — which is what lets two parties bill against the meter without trusting each other's logs. This framework unifies the fee surfaces of Section 7.1: anchoring fees, metered-utilization fees, and settlement fees all draw demand from the same source — operations on packaged data — and all scale with ecosystem activity rather than with speculation.

11 Future Works

As ROKO Network continues to evolve, several promising areas have been identified for further research and development. Consistent with this document's commitment to honest scoping, this section also delineates current implementation status: the network is in a gated-testnet phase, the verification, provenance, and anchoring machinery described in Sections 3 and 8 is built and tested, and the items below constitute the active and future build.

11.1 The On-Chain Data Ladder The anchoring posture today is *hashes now, payloads by exception, later*. Each subsequent rung is strictly additive. **Phase A — Anchor hashes** (nearest): artifact-version hashes, attachment hashes, and shard manifest checksums anchored as receipts, with provenance records gaining on-chain receipt references. **Phase B — Manifests on-chain:** publishing the shard manifest itself — version, counts, per-section checksums, source instance — as on-chain data: a public, verifiable table of contents whose bulk data remains off-chain; requires an on-chain data-payload capability on the ROKO side. **Phase C — Select shard payloads:** small, critical Knowledge Shards — registries, trust data, key indexes — published whole on-chain, envelope-encrypted when private. **Phase D — Key-material shards:** threshold-split recovery material published to chain under policy; deliberate new cryptographic design work, distinct from FROST signing shares, which never leave participant devices by design.

11.2 Escrow and Controlled Exchange Day-one settlement is direct pay-on-verified-completion. A Phase 2 escrow primitive — held funds with staged, receipt-conditioned release, and controlled data/service exchange — will serve use cases requiring conditional delivery, added only when demand warrants the additional custody surface.

11.3 Signed Metered Claims The execution substrate emits rich duration telemetry (timestamped lifecycle events, latency histograms) but does not yet package it as signed, priced "this took N metered time-units" claims. Producing those claims is the integration seam that activates time-network metering (Section 7.1, mechanism 3) as a billable product.

11.4 Public Attestation Services The stack's attestation primitives — identity, executor conformance, work completion — will be productized as paid, publicly auditable network services: a registry of claims any counterparty can check before transacting.

11.5 Consensus-Consequence Enforcement Time-quality offences are defined, detected, and recorded on-chain today; slashing enforcement and time-quality influence on block-production eligibility are being enabled in stages as the mesh matures and its parameters are validated under production load. Enabling enforcement is a runtime configuration change over machinery that already exists, not new construction.

11.6 Mainnet Launch and Platform Hardening The path from gated testnet to production includes cutting the production mainnet genesis and assigning its EVM chain identity, promoting the temporal precompile into the mainnet runtime, standardizing the staking precompile's ABI surface, and progressive decentralization of the remaining root-key posture (Section 5). The network's economic parameters will be continuously re-calibrated against observed demand as production workloads come online, with ongoing modeling closing the loop between adoption, native-coin demand, price, and the security budget. Isolation hardening of the reference execution substrate and maturation of the real-time edge-controls architecture (Section 9.3) proceed in parallel.

12 Conclusion

ROKO Network represents a purpose-built answer to a structural problem the machine economy — industrial, robotic, and agentic alike — cannot solve for itself: no participant in a machine-work transaction can credibly certify its own time, its own occurrence claims, or its own settlement. By combining a battle-tested consensus foundation with Proof of Accurate Time — a validator time mesh whose measurements land on-chain and feed back into consensus — signed temporal receipts on every transaction, deterministic ordering with enforced inclusion, and independently re-verifiable anchoring receipts, ROKO supplies the guarantees that must come from a neutral network. Through the TemporalFeePool, it funds the accuracy that makes those guarantees worth buying.

The Temporal EVM platform makes these capabilities available to every existing Ethereum developer with no new tooling, while the Traceable Work Settlement Layer positions ROKO within an open, standards-based stack — A2A execution contracts, W3C PROV provenance, content-addressed packaging, FROST-MPC custody — in which every layer is independently verifiable and the chain occupies the one seat a centralized stack structurally cannot fill. The unified metering framework extends settlement from completed work to the full data lifecycle, and the network's economics route fees to the validators whose accuracy is the product.

As the world moves deeper into an AI-driven economy, the volume of machine work — on factory floors, across vehicle fleets, and between autonomous agents — performed between mutually distrusting parties will only grow — and with it, the need for a neutral clock, notary, and clearing layer. ROKO Network is built to be that layer. These collective capabilities invite the broader open-source community to participate in this journey toward a traceable, verifiable, and economically settled machine-work future.

Partners & Acknowledgements

This work is carried out with a set of independent partners and service providers, acknowledged with thanks.

Integro Labs, LLC — ROKO Network's technology development and leadership partner; AI orchestration and custom automation solutions. https://integrolabs.io

Unforkable — DeFi engineering specialists building secure smart contracts and full-stack solutions. https://unforkable.co

Hyperspawn Robotics — low-cost robotics systems: manufacturing, testing, and deploying experimental control policies for the open-source DROPBEAR biped; co-developer of the real-time edge-controls architecture (Section 9.3). https://www.hyperspawn.co

Fractional Robots — think tank and rapid prototyping focused on the development of AI-powered software and hardware. https://fractionalrobots.com

Selfient, Inc. (operating commercially as **ROSÉ**) — independent enterprise software company providing cryptographic proof of AI execution and compliance infrastructure for regulated industries. https://www.selfient.xyz

TimeBeat (service provider) — end-to-end timing and IEEE 1588 clock-synchronization ecosystem provider. https://www.timebeat.app

Iskout (service provider) — rapid precision hiring and talent acquisition for technology companies. https://www.iskout.com

The network is built on the Polkadot SDK [13] and the Open Compute Project Time Appliances Project (OCP TAP), whose IEEE 1588 PTP timing infrastructure [15] the time mesh interoperates with. Finally, thanks are owed to the **ROKO community** — the supporters and longstanding community members who have made all of this possible.

References

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

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

[3] S. Haber and W. S. Stornetta, "How to Time-Stamp a Digital Document," *Journal of Cryptology*, vol. 3, no. 2, 1991.

[4] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)," RFC 3161, August 2001. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc3161

[5] D. Mills, J. Martin, J. Burbank, and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification," RFC 5905, June 2010. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc5905

[6] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," *Communications of the ACM*, vol. 21, no. 7, 1978.

[7] L. Moreau and P. Missier (eds.), "PROV-DM: The PROV Data Model," W3C Recommendation, April 2013. [Online]. Available: https://www.w3.org/TR/prov-dm/

[8] A2A Project, "Agent2Agent (A2A) Protocol," Linux Foundation open-source project, contributed by Google. [Online]. Available: https://github.com/a2aproject/A2A

[9] Y. Rao, "Bittensor: A Peer-to-Peer Intelligence Market." [Online]. Available: https://www.bittensor.com/whitepaper

[10] A. Langley, M. Hamburg, and S. Turner, "Elliptic Curves for Security," RFC 7748, January 2016. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7748

[11] M. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," NIST SP 800-38D, November 2007.

[12] C. Komlo and I. Goldberg, "FROST: Flexible Round-Optimized Schnorr Threshold Signatures," *Selected Areas in Cryptography (SAC)*, 2020.

[13] G. Wood, "Polkadot: Vision for a Heterogeneous Multi-Chain Framework," 2016. [Online]. Available: https://assets.polkadot.network/Polkadot-whitepaper.pdf

[14] A. Stewart and E. Kokoris-Kogias, “GRANDPA: A Byzantine Finality Gadget,” 2020. [Online]. Available: https://arxiv.org/abs/2007.01560

[15] IEEE, “IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems,” IEEE Std 1588-2019.

[16] A. Armanni and G. Lattanzi, “Roko DAO Governance: Review and System Improvements,” Exa Group, commissioned report, September 2024.

[17] A. Armanni and G. Lattanzi, “Tokenomics Review and Sustainable Economic Model,” Exa Group, commissioned report, September 2024; and ROKO Network, “Roko Tokenomics R&D,” internal working document.

[18] A. Armanni and G. Lattanzi, “Roko DAO Treasury Management Strategy,” Exa Group, commissioned report, September 2024.

[19] A. Armanni and G. Lattanzi, “Roko Network: Product and Market Positioning,” Exa Group, commissioned report, August 2024.

[20] ROKO Network, “Roko Time Node — Proof of Capability,” design specification.

[21] ROKO Network, “ROKO Value Framework,” internal report, 2026 — documents the EGG edge gateway, the DROPBEAR open-source biped (with Hyperspawn Robotics and Point Blank), and robotics-orchestration positioning.