

Title: Blockchain-Anchored Compute Network for Public-Benefit Tasks with a Proof-of-Processing Validation Layer, Accountability-Scaled Verification, and an Environmentally Weighted Incentive Mechanism

“REVO” or “Project: REVO”

Created by Blake Christopher Curtis

1. Field of the Invention

The present disclosure relates to decentralized computing networks and distributed ledger (blockchain) systems. More particularly, it relates to a network that routes useful compute jobs (for example, artificial-intelligence inference, scientific simulation, data analytics and model training) to heterogeneous compute nodes; verifies that the work was performed correctly through a Research and Processing Validation Layer ("Proof-of-Processing" or "PoP"); scales the intensity of that verification to the physical accountability of each node; records tamper-evident compute receipts on a ledger; and weights node incentives by verified environmental behavior through a sustainability score ("VerdantScore").

2. Background

Conventional proof-of-work blockchains expend large amounts of computation and electricity on arbitrary puzzles whose only purpose is to secure the ledger. At the same time, demand for useful computation (for example, AI inference and scientific simulation) is growing rapidly, and the electricity and grid connections needed to build new centralized data centers have become a binding constraint. Large amounts of existing computing capacity in homes, laboratories, universities and enterprises sit idle or under-used.

Networks that attempt to use distributed capacity face several unsolved problems. First, a requester cannot easily confirm that a remote, untrusted machine actually performed a computation correctly without repeating the computation in full. Second, unlike a proof-of-work puzzle, which is expensive to solve but cheap to check, useful computation is typically expensive both to perform and to check. Third, the outputs of many workloads, particularly AI inference on graphics processors (GPUs), are not bit-for-bit reproducible across different hardware models, so naive comparison of output hashes flags honest work as fraudulent.

Trusted execution environments (TEEs) and remote attestation have been proposed as a way to establish trust in remote machines. However, publicly reported research has demonstrated that a party with physical access to a machine can defeat memory protections of widely deployed TEEs using inexpensive hardware placed between the processor and memory, and in some cases can forge attestation reports. Chip vendors have generally treated such physical attacks as outside their threat model. In a decentralized compute network, the operator of a node has physical access

to that node and may have a financial incentive to cheat, so attestation alone is not a sufficient basis for trust.

Further, claims that computation is powered by clean energy are commonly supported by annual, unbundled certificates that are not matched in time or location to the electricity actually consumed, and software-reported energy figures can be spoofed. There is therefore a need for a network that (a) delivers verifiable, public-benefit computation; (b) pays out only after proof of correct processing; (c) calibrates how much it relies on any single trust signal, including hardware attestation, to the real-world accountability of the party controlling the hardware; (d) records compact, privacy-preserving, independently checkable receipts; and (e) adjusts rewards according to measured and cross-checked environmental impact.

3. Summary of the Invention

The disclosure provides a blockchain-anchored compute network comprising three coupled layers: (1) a Job Orchestration Layer that accepts, classifies, partitions and routes jobs and supports three funding and priority lanes; (2) a Research and Processing Validation Layer (PoP) that verifies results; and (3) an Environmentally Weighted Incentive Mechanism ("VerdantScore") that computes a node's sustainability score and applies it to rewards and routing.

Aspects of the disclosure include, without limitation, the following, each of which may be used alone or in combination:

- Commit-before-compare verification, in which each node commits to a salted hash of its output before any node's output is revealed, preventing a node from copying another node's answer.
- Two comparison modes selected per job class: an exact-match mode for deterministic workloads and a tolerance mode for non-deterministic workloads, in which outputs are revealed privately to assigned verifiers only after all commitments are recorded and are accepted if within a declared tolerance.
- Merkle-committed spot-check verification, in which a node commits to a Merkle root over hashed intermediate checkpoints, verifiers randomly select checkpoints after the commitment, and the node must open each selected checkpoint with an inclusion proof.
- Accountability-scaled verification, in which each node is assigned an accountability level based on identity verification, contractual undertakings, physical control of the hardware and audit rights; the frequency and depth of verification and the weight given to hardware attestation are set according to that level; verification never falls to zero at any level; and sensitive-data jobs are routed only to nodes at or above a threshold level.
- Verifier-selection safeguards that select verifiers primarily by stake and randomness, cap or exclude environmental-score influence on verifier selection, and exclude verifiers sharing an operator, stake source or network cluster with the node being verified.

- Separation of ledger-writing from environmental scoring, in which validators that order and finalize ledger entries are selected by stake, and the environmental score is used for compute assignment, rewards and capped verifier weighting rather than validator selection.
- A structured compute receipt with canonical encoding, salted commitments, a signature domain binding the receipt to a specific ledger and contract, explicit labelling of unverified runtime metadata as claims, and on-ledger checking that a recovered signer matches the node and verifiers registered for the job.
- Batch anchoring of receipts under a Merkle root with distinct leaf and internal-node hashing, batch intervals that differ by funding lane, and delivery of inclusion proofs to both the submitter and the node at the time of anchoring.
- Trust-scaled energy evidence, in which meter readings are signed by the meter's own secure element, weighted by the accountability of the party physically controlling the meter, and cross-checked against grid data and the machine's reported load.
- Key separation for node operators (a receipt-signing key distinct from the key controlling earnings and stake) and multi-party control of treasury funds by independent keyholders.
- Forced inclusion, by which a submitter whose job is ignored by the orchestrator may submit the job directly to the ledger, together with published stages for decentralizing the orchestrator.

Token rewards follow a fixed-supply emission schedule with a year-capped "green budget" that reallocates environmental bonuses within the annual cap and burns unused amounts rather than minting new supply. Sensitive data executes only in qualifying node pools; the ledger stores hashes and receipts, not the underlying data.

4. Brief Description of the Drawings

FIG. 1 — Block diagram of the system architecture.

FIG. 2 — Flow diagram of the job lifecycle from submission to payout.

FIG. 3 — Commit-before-compare verification with exact and tolerance comparison modes.

FIG. 4 — Merkle-committed spot-check verification for long-running jobs.

FIG. 5 — Accountability ladder and accountability-scaled verification.

FIG. 6 — Verifier selection with safeguards.

FIG. 7 — Compute receipt structure and batch anchoring.

FIG. 8 — Separation of ledger-writing (validator selection) from environmental scoring.

FIG. 9 — VerdantScore pipeline with trust-scaled energy evidence.

FIG. 10 — Three funding and priority lanes.

FIG. 11 — Key separation for node operators and treasury.

FIG. 12 — Reward computation. [ASCII]

FIG. 13 — Data trust and job safety pipeline. [ASCII]

FIG. 14 — Storage topology. [ASCII]

FIG. 15 — Governance hardening. [ASCII]

FIG. 16 — Annual emissions cap and green budget. [ASCII]

Figures (Key Graphics)

FIG. 1 — System Architecture

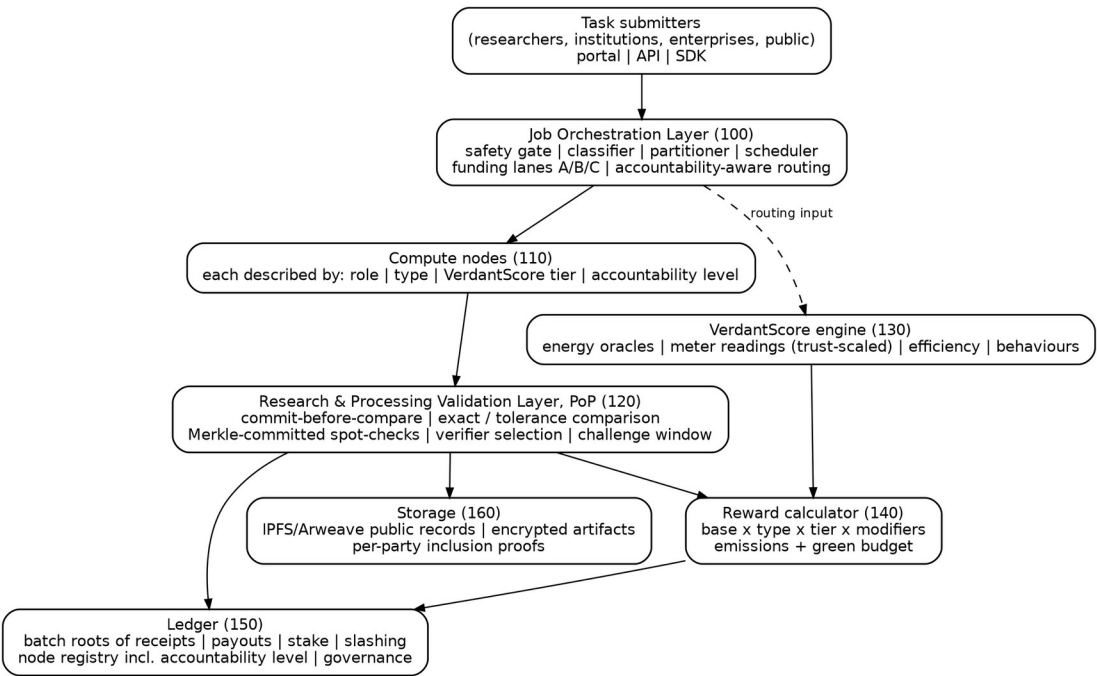


FIG. 2 — Job Lifecycle

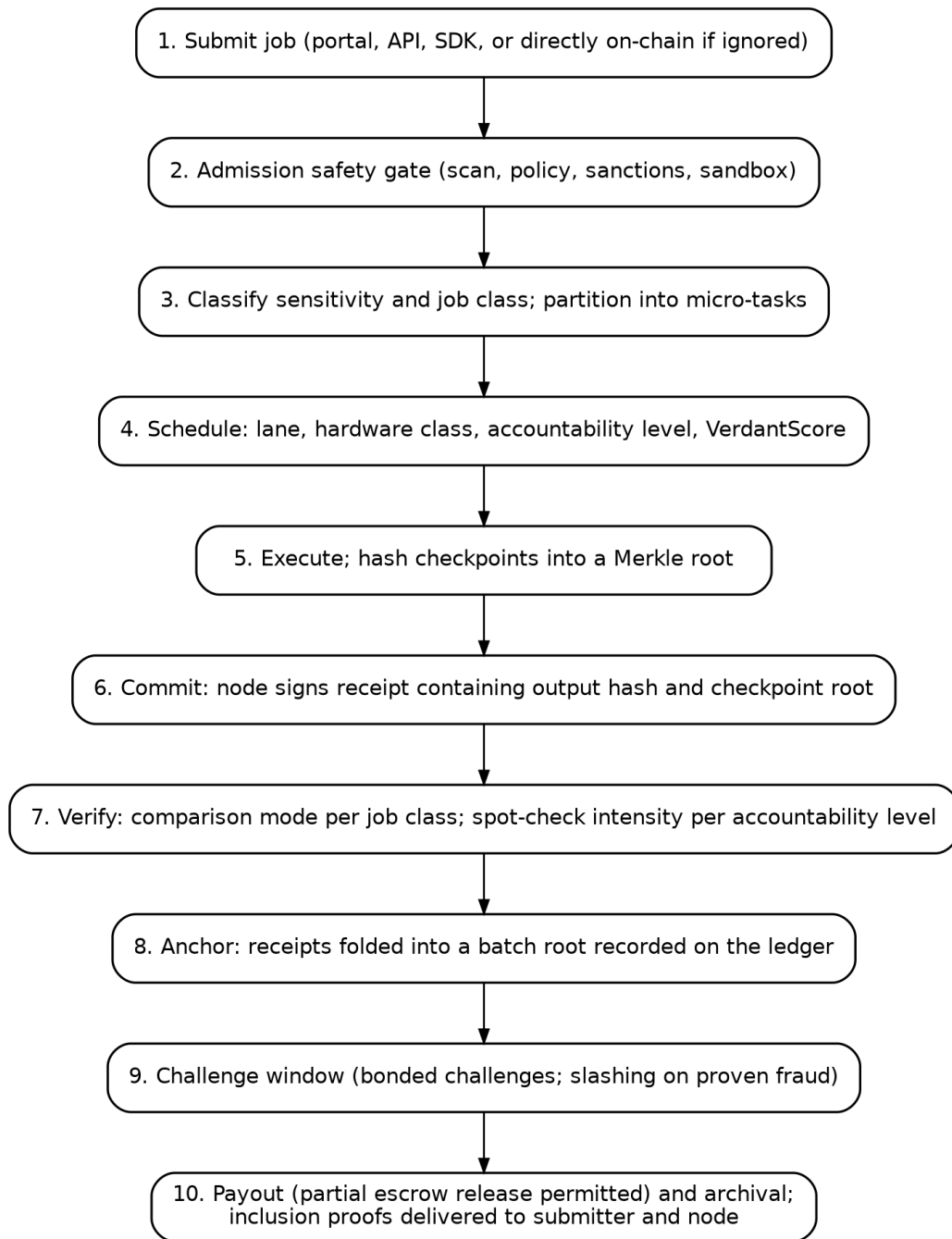


FIG. 3 — Commit-Before-Compare with Two Comparison Modes

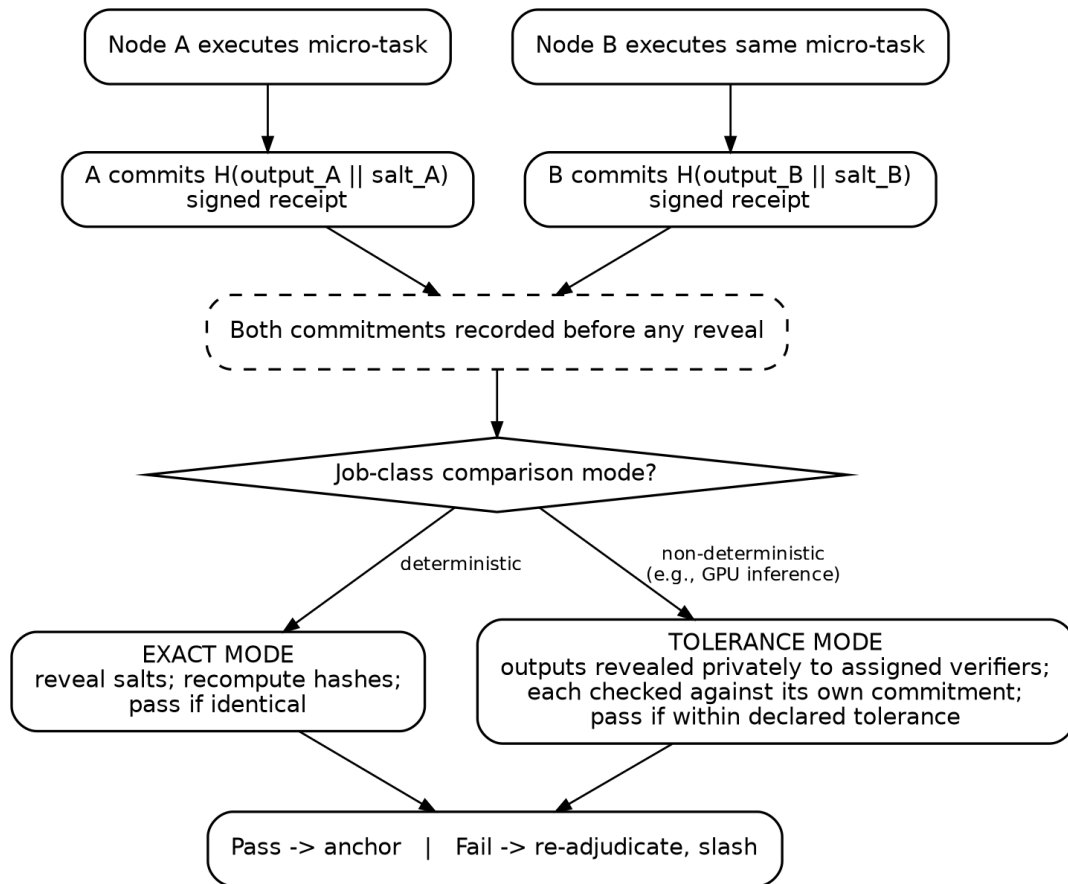


FIG. 4 — Merkle-Committed Spot-Check Verification

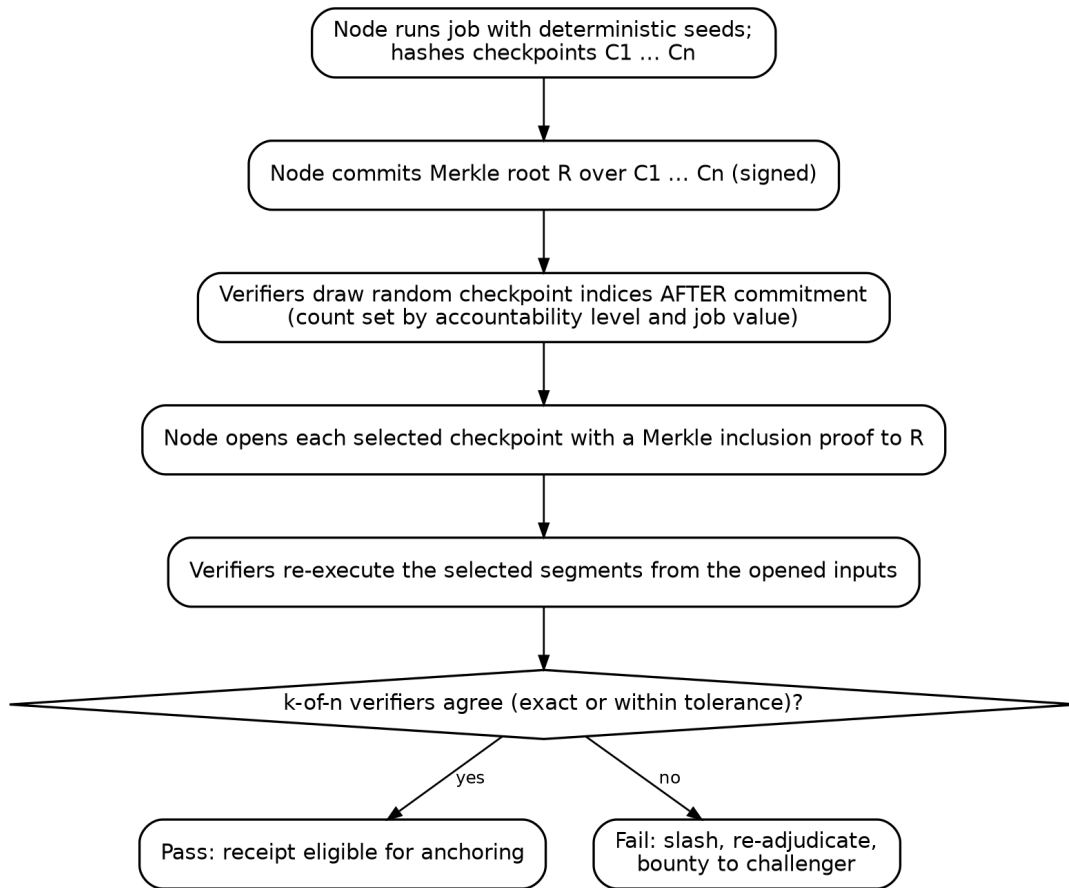


FIG. 5 — Accountability Ladder

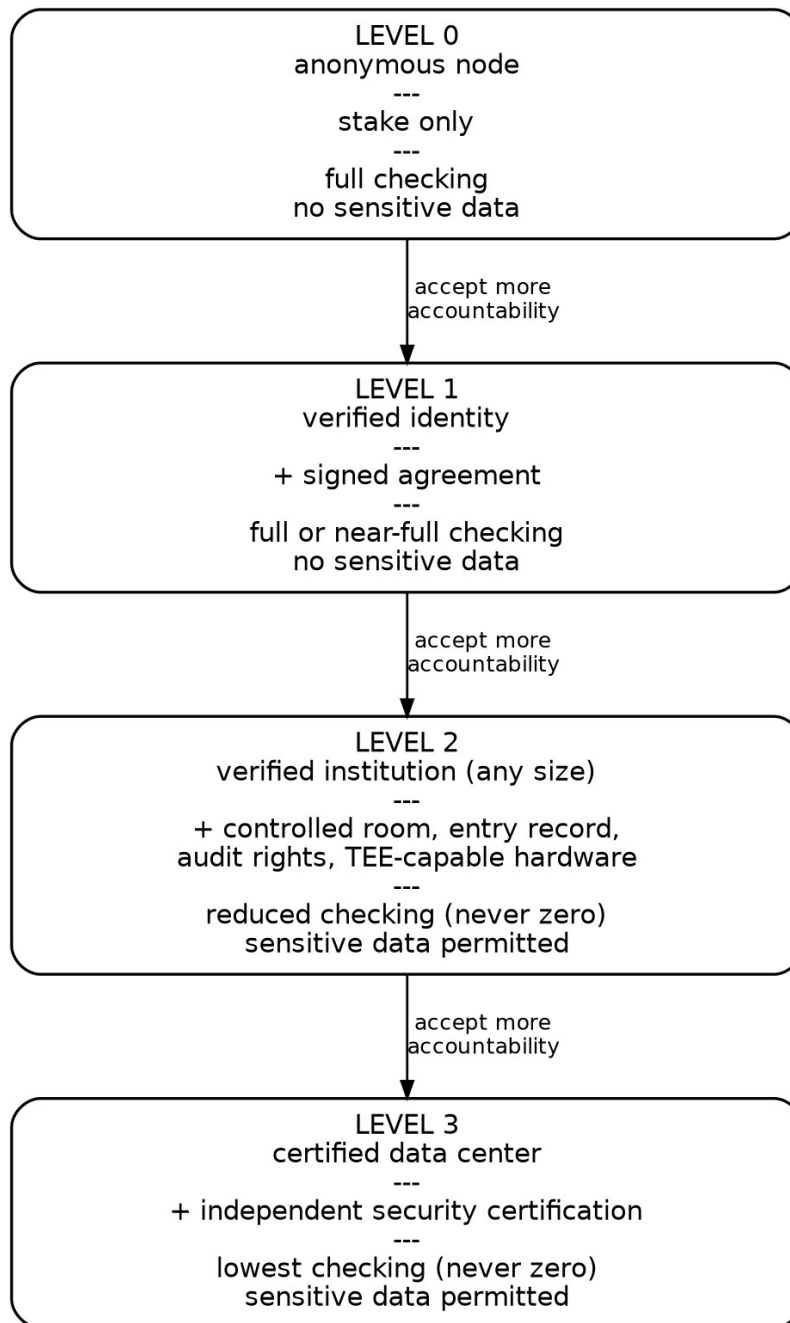


FIG. 6 — Verifier Selection Safeguards

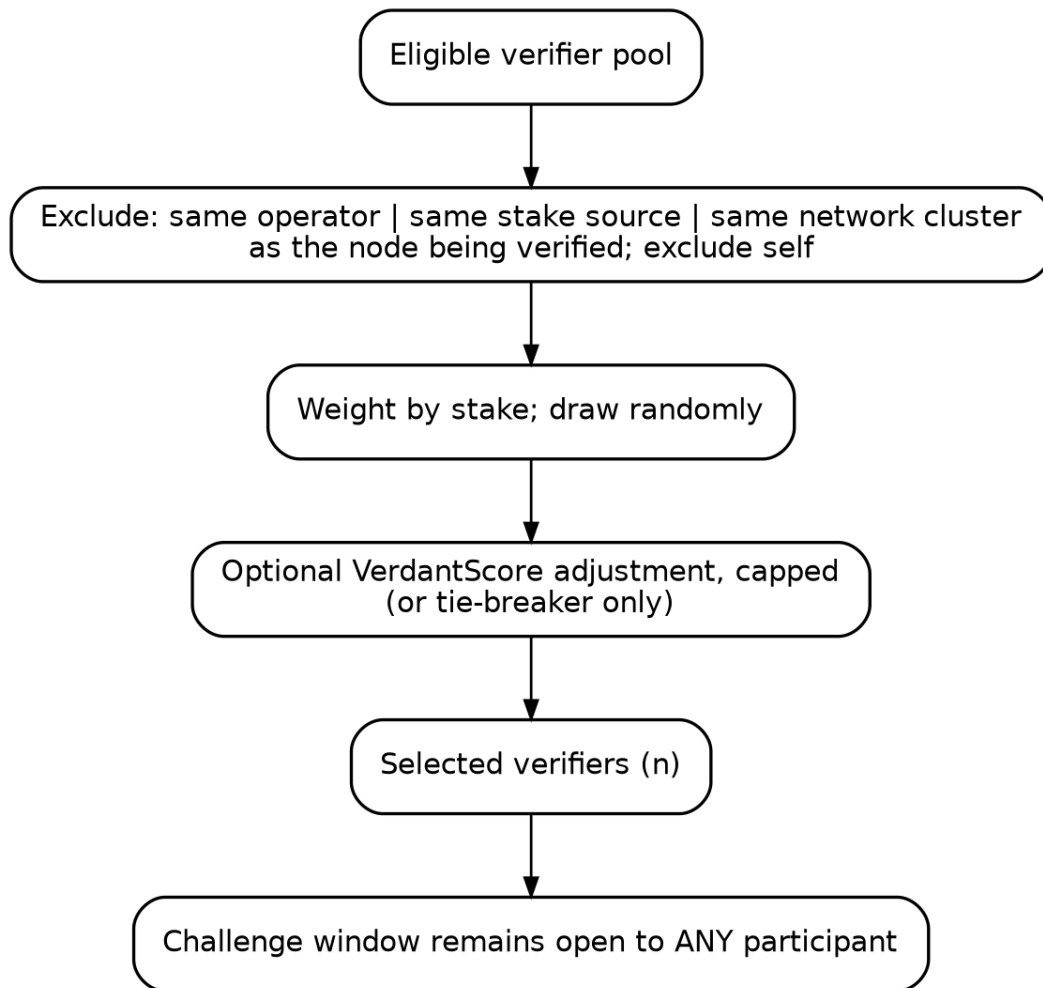


FIG. 7 — Compute Receipt and Batch Anchoring

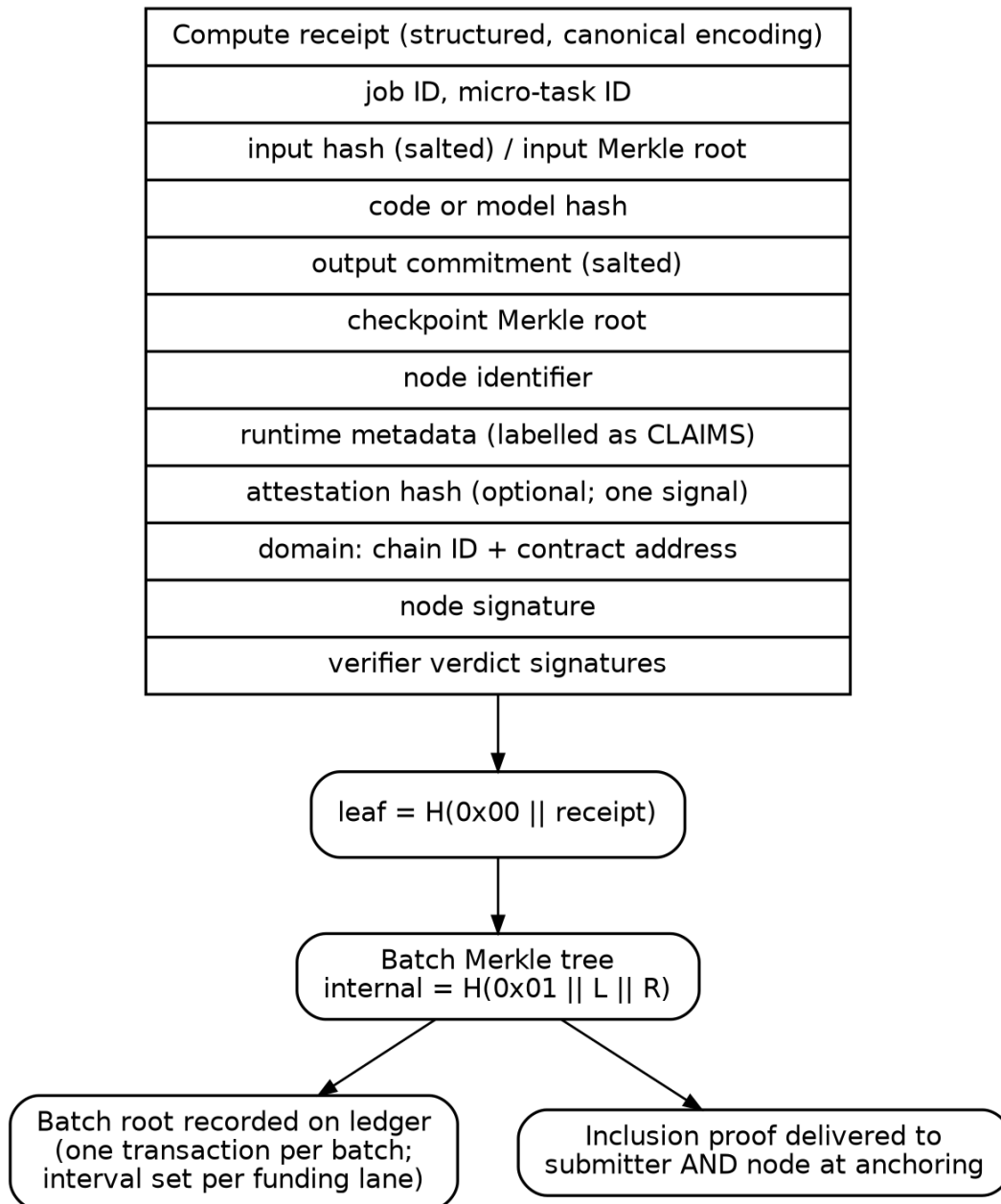


FIG. 8 — Ledger-Writing Separated from Environmental Scoring

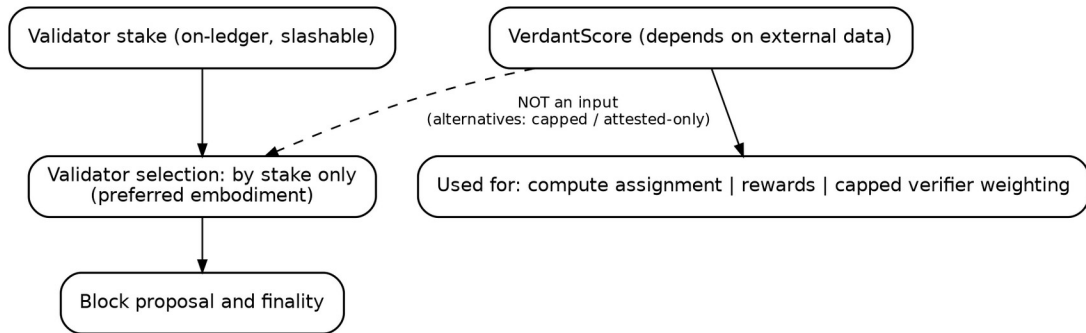


FIG. 9 — VerdantScore Pipeline

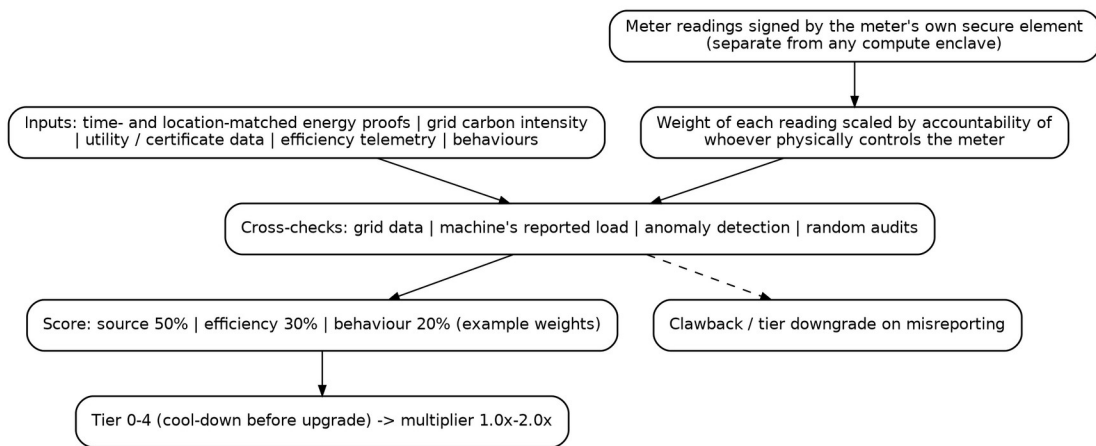


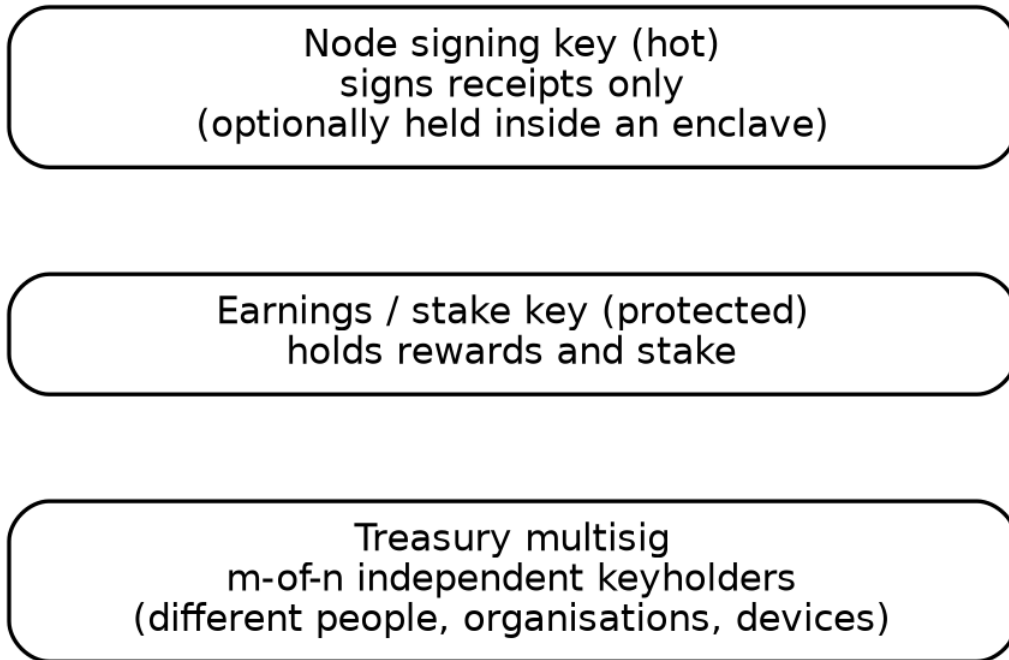
FIG. 10 — Three Funding/Priority Lanes

LANE A: Protocol-funded (default)
nodes paid from emissions/treasury
submitter pays nothing; may need no wallet
slower, lower-cost anchoring batches

LANE B: Enterprise staked-priority
stake as collateral; queue priority
nodes still paid by protocol

LANE C: Direct-pay boost
optional bounty on a job
added to base reward

FIG. 11 — Key Separation



5. Detailed Description

The following description sets out example embodiments. The examples, values, percentages, thresholds and names are illustrative and are not intended to limit the scope of the disclosure. Features described in one embodiment may be combined with features of other embodiments. The terms "may," "can," "for example," and "in some embodiments" indicate optional features.

5.1 Entities and Roles

- Task submitter: a researcher, institution, nonprofit, enterprise or member of the public who submits a job.
- Compute node: a participant device that performs computation. A compute node may be a laptop, a desktop or gaming computer, a laboratory workstation, a multi-GPU server, a university cluster or a data-center rack.
- Verifier: a node or committee of nodes that checks results, for example by re-executing selected portions of a computation or comparing outputs.
- Validator: a node that orders, validates and finalizes ledger entries, in embodiments in which the network operates its own ledger or rollup.

- Metering device: an optional device that measures the electricity consumed by a compute node and signs its readings.
- Orchestrator: the component that admits, partitions, schedules and routes jobs.
- Energy oracle: an external data feed providing time- and location-matched information such as grid carbon intensity, renewable sourcing or utility certification.
- Ledger and governance: on-ledger recording of receipt roots, payouts, stake, slashing, the node registry and governance decisions ("The REVO Accord").
- Storage services: decentralized storage (for example IPFS or Arweave) for immutable public records, and encrypted off-ledger storage for large or sensitive artifacts.

In some embodiments, roles are separated so that no participant judges its own work: a verifier is never assigned to verify its own output or that of an affiliated node, and a validator does not approve its own payout. A single machine may hold more than one role only where the roles do not permit it to judge its own work.

5.2 Node Descriptors

In some embodiments each node is described by four independent descriptors that are recorded in a node registry:

- Role: compute, verifier, validator and/or metering device, as described in section 5.1.
- Type: for example General (example multiplier 1.0x), GPU (1.25x), Verified Research (1.5x) and Enterprise (1.75x to 2.0x). Type is established by benchmark tests at onboarding and spot-checked afterwards, rather than self-declared. Hardware attestation, where available, may support but does not replace benchmarking.
- VerdantScore tier: for example Tier 0 to Tier 4, with reward multipliers from 1.0x to 2.0x (section 5.11).
- Accountability level: for example Level 0 to Level 3 (section 5.7), which governs verification intensity and eligibility for sensitive-data jobs.

Hardware class (for example, available GPU memory and memory bandwidth) determines which jobs a node is capable of running, while the descriptors above determine routing, verification intensity and rewards. A node may comprise multiple tightly interconnected processors; communication-intensive work may be routed to a single node large enough to hold it, or to a pool of physically co-located nodes, because communication limits apply between nodes rather than within them.

5.3 Job Lifecycle

Referring to FIG. 2, a job proceeds through: submission; admission through a safety gate; classification by sensitivity and job class and partitioning into micro-tasks; scheduling; execution; commitment; verification; anchoring; a challenge window; payout; and archival with delivery of inclusion proofs. Micro-tasks carry resource tags (for example CPU or GPU needs and memory requirements) and a sensitivity class. Scheduling considers the funding lane, impact tags, fairness, hardware class, accountability level and VerdantScore.

In preferred embodiments the network prioritizes batch, independent and delay-tolerant workloads, such as batch AI inference and scientific simulation (for example climate ensembles, molecular docking, genomics quality control and Monte Carlo analysis), which divide into independent pieces and can be scheduled at times and places where cleaner electricity is available. Real-time workloads requiring sub-second responses may be excluded. Fine-tuning on a single node and multi-node training may be supported in further embodiments.

Table 1 - Example job categories

Category of job	Example	Output
AI inference (batch)	Run a protein-structure model on new sequences; classify medical images	Predictions and confidence scores
Scientific simulation	Climate nowcasting or epidemic-spread modelling	Forecast data and scenarios
Data analytics	Genomic sequence analysis or risk modelling	Statistical results, patterns, risk profiles
AI training / fine-tuning	Training or fine-tuning a tumor-detection model	Updated model
Rendering / visualization	Large 3D or molecular rendering	Rendered images or models

Table 1 above is illustrative. The examples are not intended to limit the scope of the disclosure.

5.4 Job Submission and Enterprise Integration

Jobs may be submitted manually through a web dashboard or automatically through an application programming interface (API) or software development kit (SDK) that streams jobs from existing institutional or enterprise systems. For example, a cancer research institute may connect its imaging pipeline so that each new batch of scans is submitted automatically; a financial firm may submit portfolio scenario simulations on a recurring schedule.

The network may maintain pre-approved, always-available job pools (a "baseline job book"), for example climate nowcasting simulations or biomedical inference on public datasets, so that nodes have meaningful work even when external submissions are slow. The governance body may

approve additions to the baseline job book, for example a language-preservation translation workload submitted by a nonprofit, routed to smaller General nodes.

In some embodiments, a submitter in the protocol-funded lane, who pays nothing, is not required to hold a cryptocurrency wallet or private key; the submitter authenticates through a conventional login and the network performs ledger operations on the submitter's behalf. In some embodiments, the network or an application sponsors transaction fees ("gas sponsorship") so that paying customers need not hold the host ledger's native coin.

5.5 Research and Processing Validation Layer (Proof-of-Processing)

The validation layer verifies that work was performed correctly before rewards are released. It may combine the following techniques, selected per job class and per accountability level.

Commit-before-compare (FIG. 3). When two or more nodes execute the same micro-task for redundancy, each node first submits a signed commitment consisting of a hash of its output combined with a random salt. Outputs and salts are revealed only after all commitments for that micro-task are recorded. Because a node cannot see another node's output before committing, it cannot copy another node's answer and claim credit for work it did not perform.

Two comparison modes (FIG. 3). Each job class is assigned a comparison mode. In an exact-match mode, used for deterministic workloads, nodes reveal their salts and the network recomputes and compares the committed hashes; the result passes if the hashes are identical (or a majority match). In a tolerance mode, used for workloads whose outputs legitimately vary across hardware (for example GPU inference, where floating-point arithmetic differs slightly between hardware models), outputs are revealed privately to the assigned verifiers, for example over encrypted channels, after all commitments are recorded; each revealed output is checked against its own commitment; and the result passes if the outputs agree within a declared tolerance appropriate to the job class. In some embodiments, outputs are compared only between nodes of the same hardware class to narrow the tolerance.

Merkle-committed spot-checks (FIG. 4). For long-running or heavy jobs, a node runs with deterministic seeds where possible, hashes periodic checkpoints of intermediate state, and commits a Merkle root over those checkpoint hashes in a signed receipt. After the commitment, verifiers draw random checkpoint indices using a source of randomness not known to the node in advance. The node must open each selected checkpoint with a Merkle inclusion proof tying it to the committed root, and verifiers re-execute the corresponding segments (for example random mini-batches, layers or windows of steps) from the opened inputs. A result passes if at least k of n verifiers agree, exactly or within tolerance. Because the node cannot know which checkpoints will be selected, faking any appreciable fraction of the work carries a high probability of detection; for example, if 10 percent of segments are fabricated and 20 segments are sampled independently, the probability of detection is approximately 88 percent. The number of samples is set according to job value and the node's accountability level, and slashable stake is sized so that the expected loss from detection exceeds the expected gain from cheating.

Attestation as one signal. Where a node has a trusted execution environment, the node may attach a remote-attestation report (or a hash of it) and signed execution logs to its receipt. The weight given to attestation is set by the node's accountability level (section 5.7). Attestation alone does not substitute for verification at any level.

Challenge window. After verification, a time-boxed challenge window opens during which any participant may post a bond and challenge a result by re-executing designated samples against the committed roots. A demonstrated mismatch triggers slashing of the responsible node's stake (and, where applicable, of verifiers who approved it), re-adjudication, and a bounty to the challenger; a failed challenge forfeits some or all of the bond. Only one honest challenger is needed to expose fraud. The window length may be set per job class (for example, from hours to days) so that it exceeds the time a challenger needs to re-execute sampled segments.

Payout gating. Rewards unlock after the challenge window closes, or in part earlier through partial escrow release while the node's stake remains locked until the window closes.

5.6 Verifier Selection Safeguards

Referring to FIG. 6, verifiers are drawn from an eligible pool after excluding the node being verified and any node sharing its operator, stake source or network cluster. Remaining candidates are weighted by stake and drawn randomly. In some embodiments an environmental-score adjustment is applied to verifier weighting but capped at a small fixed amount, or used only to break ties, so that falsified environmental data cannot materially increase a party's chance of verifying its affiliates' work. The challenge window remains open to any participant regardless of selection.

5.7 Accountability-Scaled Verification

Referring to FIG. 5, each node is assigned an accountability level reflecting how accountable the party physically controlling the hardware is, independent of the size of the operator.

Accountability may be established by one or more of: verified identity of the operator; a signed agreement with the network including audit rights and consequences for misconduct; physical control measures such as a locked room, a record of who enters, and video monitoring; independent security certification of the facility; and stake. An example ladder is:

Table 2 - Example accountability ladder

Level	Who	What establishes it	Verification	Sensitive data
0	Anonymous node	Stake only	Full verification intensity	Not permitted
1	Verified individual or organization	Verified identity, signed agreement, stake	Full or near-full	Not permitted

2	Verified institution of any size (for example a university laboratory or a hospital server room)	Level 1 plus controlled physical space, audit rights, and TEE-capable hardware	Reduced, never zero	Permitted
3	Certified professional data center	Level 2 plus independent security certification	Lowest, never zero	Permitted

Table 2 above is illustrative. The examples are not intended to limit the scope of the disclosure.

The orchestrator and validation layer set, according to the node's level: (a) the fraction of micro-tasks subjected to redundant execution; (b) the number of spot-check samples; (c) the weight given to hardware attestation; and (d) eligibility for sensitive-data jobs, which may require both a threshold level (for example Level 2) and hardware supporting confidential computing. At every level a non-zero rate of random verification is maintained, so that even highly accountable nodes, including insiders at accountable facilities, remain subject to detection. Nodes may move between levels upon meeting or ceasing to meet the criteria, upon audit findings, or upon proven misconduct.

The rationale is that a party with physical access to hardware may be able to defeat or forge hardware attestation at low cost; the practical risk of such an attack depends on whether the party is identifiable, contractually bound, physically supervised and legally exposed. Scaling reliance on attestation and the intensity of verification to accountability recovers much of the cost benefit of trusted hardware where trust is earned, without making the security of the network depend on any single chip. The same principle may be applied to metering devices (section 5.11).

5.8 Compute Receipts

Referring to FIG. 7, a compute receipt is a structured record for a micro-task that may include: a job identifier and micro-task identifier; a hash of the inputs (salted where inputs are sensitive or guessable) and, for large inputs, a Merkle root over input chunks; a hash identifying the code or model executed; a salted commitment to the output; a Merkle root over checkpoint hashes; the node identifier; runtime metadata such as hardware class, start and end times and grid region; an optional attestation hash; a signature domain; the node's signature; and verifiers' verdict signatures.

- Canonical encoding: receipts are serialized in a single defined byte layout so that honest nodes producing the same logical result produce identical hashes.
- Signature domain: the signed content includes an identifier of the ledger (for example a chain identifier) and of the receiving contract, so that a receipt cannot be replayed on another ledger or

contract. On Ethereum-compatible ledgers this may use typed structured-data signing (for example EIP-712); equivalent mechanisms may be used on other ledgers.

- Signer verification: the receiving contract recovers or verifies the signer of each signature and checks that it matches the node and verifiers registered for that job, rejecting the receipt otherwise.
- Replay protection: the job identifier, micro-task identifier, timestamp and signature domain together prevent a valid receipt from being resubmitted for additional payment.
- Claims labelling: runtime metadata and energy data that are not proven by the receipt itself are labelled as claims, so that recording on the ledger is not mistaken for verification.
- Privacy: only hashes, roots and signatures are recorded on the ledger; inputs and outputs remain off-ledger, encrypted where sensitive.

5.9 Batch Anchoring and Data Availability

Rather than recording each receipt in its own ledger transaction, receipts are folded into a batch Merkle tree and only the batch root is recorded, so that a batch of any size is anchored in a single transaction. In some embodiments leaves and internal nodes are hashed with different prefixes (for example a leaf is $H(0x00 \parallel \text{receipt})$ and an internal node is $H(0x01 \parallel \text{left} \parallel \text{right})$) to prevent an internal node from being presented as a receipt. The batch interval may differ by funding lane, for example shorter intervals for paid lanes and longer, lower-cost intervals for the protocol-funded lane. Ledger fees for anchoring protocol-funded receipts may be budgeted explicitly within the emissions model.

At the time of anchoring, an inclusion proof for each receipt is delivered to both the submitter and the node, so that either party can later prove that its receipt was recorded using only its receipt, its proof and the on-ledger root, without revealing other parties' receipts and without depending on the network operator remaining available. Full batch trees are retained in decentralized and/or encrypted storage to preserve availability.

5.10 Consensus and Validator Selection

Referring to FIG. 8, in embodiments in which the network operates its own ledger or rollup, validators that order and finalize ledger entries are selected by stake, for example in proportion to stake with random selection, with slashing for provable misbehavior such as signing conflicting entries. In a preferred embodiment the environmental score is not an input to validator selection, because the score depends on external data that can be falsified, and allowing it to add voting weight could allow a party to exceed a safety threshold (for example one third of voting weight) without committing additional stake. The environmental score is instead used for compute assignment, reward multipliers and capped verifier weighting. In alternative embodiments,

environmental or reputation inputs may affect validator weight only within a small fixed cap, or only when backed by long-held, attested scores and slashable stake.

The disclosure is not limited to a particular ledger. In embodiments the network may operate as a token and smart contracts on an existing public ledger or layer-2 network; as its own rollup settling to a base ledger; or as an independent ledger. Receipt anchoring, signer verification, slashing of compute-node stake, the funding lanes, emissions, the green budget, governance and the node and accountability registry may be implemented as smart contracts on any such ledger.

5.11 Environmentally Weighted Incentive Mechanism ("VerdantScore")

Referring to FIG. 9, VerdantScore combines, in one example weighting, (i) energy-source verification (50 percent), (ii) energy efficiency (30 percent), and (iii) sustainability behaviors (20 percent). Tiers (for example Tier 0 to Tier 4) multiply rewards (for example 1.0x to 2.0x) and affect routing and privileges. Anti-gaming measures include time- and location-matched energy evidence, rejection of certificate-only claims lacking time and location match, scoring against marginal or hourly grid intensity rather than annual averages, cool-down periods (for example 30 days) before tier upgrades take effect, random audits, and clawbacks for misreporting.

Concept chart for the reward tiers and what they provide (V.S. = "VerdantScore")

Tier	"V.S."	Reward Multiplier	Other Benefits
Tier 0 (Baseline)	25–39	1.00x	- Basic network access - No bonuses
Tier 1 (Greenlight)	40–59	1.10x	- Priority task routing - Eligible for EcoChallenge bonuses
Tier 2 (Verdant)	60–79	1.25x	- Lower staking requirements - Voting power boost - Access to EcoMatch partnerships (matchmaking with research orgs)
Tier 3 (Evergreen)	80–94	1.50x	- Green research exclusives - Eligible for EcoRoyalties (2% cut of revenue from research jobs you support) - Free DAO proposal rights
Tier 4 (Emerald Guardian)	95–100	2.00x	- Treasury-funded grants (\$10K–\$100K scale) - Custom REVO "Green Medal" NFT (Soulbound) - Featured on REVO's homepage - Access to EcoPrime staking pools (up to 20% APY) - Option to join REVO's Green Council (advisory body)

Trust-scaled energy evidence. A trusted execution environment can attest to software but cannot measure electricity. In some embodiments, energy readings are produced by a metering device that has its own secure element and signs its readings, separate from any compute enclave, optionally with tamper-evident seals. The weight given to a meter's readings is scaled by the accountability level of the party physically controlling the meter, and readings are cross-checked against grid data, the compute node's own reported load and job history, and anomaly detection, with random audits. Readings that fail cross-checks reduce the score and may trigger clawback.

Because delay-tolerant jobs can be scheduled when and where cleaner electricity is available, and can be paused in regions where the grid is strained, the scheduler may use VerdantScore and real-time grid data to route work toward cleaner power and to provide flexible demand.

5.12 Rewards, Emissions and Modifiers

In one embodiment, $\text{payout} = \text{base_job_value} \times \text{node_type_multiplier} \times \text{tier_multiplier} \times \text{product}(\text{modifier_bonuses})$ (FIG. 12). Modifiers may include a first-validator bonus, an accuracy streak, SLA compliance and governance-approved incentives. base_job_value may be derived from measured compute for the job class subject to a declared ceiling.

Emissions follow a fixed total supply with a multi-year declining schedule for node rewards. A green multiplier budget is capped within each year and burns if unused, so that environmental bonuses reallocate within the annual cap without increasing total supply (FIG. 16). Epochs may be approximately 30 days, with rewards streamed across epochs.

Programs funded from the green budget, transaction fees or treasury may include: EcoPrime Pools (groups of high-VerdantScore nodes prioritized for sustainability-tagged jobs, paired with staking pools funded from the green budget); EcoRoyalties (recurring distributions to high-VerdantScore nodes per job and at the network level); EcoChallenges (periodic sustainability competitions open to all participants); EcoMatch (a framework connecting institutions with sustainable compute pools); and EcoBonus (one-time boosts for exceptional actions).

5.13 Funding and Priority Lanes

Referring to FIG. 10: Lane A (protocol-funded, default): the network pays nodes from emissions or treasury and the submitter need not pay or hold a wallet. Lane B (enterprise staked-priority): an enterprise stakes tokens as reputation collateral for predictable priority and routing to higher-tier pools, optionally with SLA credits; nodes are still paid by the protocol. Lane C (direct-pay boost): a submitter may attach a bounty to a specific job, added to the base reward. In some embodiments a minimum share of capacity is reserved for Lane A.

5.14 Data Trust and Job Safety

A pre-execution safety gate performs static scanning, policy classification, sanctions and export checks, and sandboxing or isolation (FIG. 13). Jobs involving protected health information or personally identifiable information execute only on nodes meeting the sensitive-data threshold of section 5.7 with confidential-computing hardware; the ledger stores only hashes and receipts; outputs are delivered over encrypted channels with access controls. Abuse triggers rejection, slashing and bans. Containers or sandboxes protect node operators from submitted code, while accountability-scaled verification and confidential computing protect submitters from node operators.

5.15 Storage and Auditability

On-ledger: compact hashes and roots for receipts, reward events, stake and governance. Decentralized storage (for example IPFS or Arweave): immutable public records such as PoP logs and governance votes. Encrypted off-ledger storage: large or sensitive inputs, outputs and logs, with retention policies by data class (FIG. 14). A public explorer may allow any party to look up receipts and verify inclusion proofs without running a full node.

5.16 Governance

Governance ("The REVO Accord") may include quiz-gated voting ("SmartVote"), delegation, quadratic caps for micro-grants, time-locks for parameter changes, and a narrowly scoped emergency council able to pause subsystems for a limited period (for example up to 72 hours) with a mandatory public report and ratification (FIG. 15).

5.17 Key Management

Referring to FIG. 11, in some embodiments each node operator holds at least two keys: a receipt-signing key that is used automatically and is authorized only to sign receipts, optionally held inside a secure enclave so that the operator cannot use it to sign arbitrary content; and a separately protected key controlling earnings and stake. Treasury funds are controlled by a multi-party (m-of-n) arrangement whose keyholders are independent people or organizations using separate devices, so that compromise of any single organization does not yield control. Private keys are generated from high-quality randomness.

5.18 Orchestrator Transparency and Forced Inclusion

In some embodiments the orchestrator is operated by an identified party in an initial phase, with correctness enforced by the validation layer and ledger rather than by the orchestrator. A submitter whose job is not scheduled within a defined period may submit the job directly to a

ledger contract, which obliges the network to schedule it ("forced inclusion"). The network may publish defined stages by which control of the orchestrator is progressively decentralized.

6. Example Embodiments

Example A (medical inference, tolerance mode). A nonprofit submits chest X-ray inference in the protocol-funded lane through a normal login, without a wallet. The job is classified as sensitive and routed to two Level 2 nodes with confidential-computing GPUs. Each node commits a salted output hash; after both commitments are recorded, outputs are revealed privately to a verifier, compared within the job class tolerance, and approved. Receipts are signed, batched and anchored; the nonprofit receives an inclusion proof. Payouts are multiplied by each node's VerdantScore tier after the challenge window.

Example B (training workload, spot-check). A laboratory fine-tunes a crop-yield model on a Level 1 node. The node hashes checkpoints every N steps and commits a Merkle root. Verifiers selected by stake and randomness, excluding affiliates, draw random checkpoint indices after the commitment; the node opens them with inclusion proofs; verifiers re-execute the windows and agree within tolerance. Because the node is Level 1, the number of samples is higher than it would be for a Level 3 node. After the challenge window passes without a successful challenge, rewards unlock.

Example C (enterprise priority). An AI lab stakes tokens in Lane B for priority routing to Level 3 nodes, without directly paying nodes; nodes are paid from emissions and treasury; the lab attaches an optional Lane C bounty to one urgent job. Paid-lane receipts anchor at a shorter batch interval.

Example D (attestation forgery contained). A Level 0 operator modifies its hardware to forge attestation reports and returns fabricated results. Because Level 0 nodes receive full verification intensity and attestation carries little weight at that level, the fabricated results fail redundant comparison and spot-checks; the operator's stake is slashed and the challenger or verifier receives a bounty.

Example E (energy claim cross-check). A node reports clean-energy consumption from a signed meter. The meter's operator is Level 1, so the reading carries moderate weight and is cross-checked against hourly grid intensity for the node's region and against the node's reported compute load. A mismatch triggers an audit and a tier downgrade with clawback.

Example F (forced inclusion). A small research group's job is not scheduled within the defined period. The group submits the job directly to the ledger contract, which obliges scheduling; the event is publicly visible.

7. Advantages Over Prior Art

- Verification of useful computation without full re-execution, by combining commit-before-compare, mode-appropriate comparison, Merkle-committed random spot-checks, staking and an open challenge window.
- Correct handling of non-deterministic workloads through a tolerance mode, avoiding false fraud findings against honest nodes.
- Security that does not depend on any single chip: reliance on hardware attestation is scaled to real-world accountability and verification never falls to zero.
- Inclusive participation: home and small operators may contribute to ordinary jobs, and small institutions may qualify for sensitive work, regardless of size.
- Ledger security insulated from manipulation of environmental data, by excluding the environmental score from validator selection.
- Compact, private, portable receipts: one ledger transaction per batch, and an inclusion proof held by each party.
- Environment-aware rewards based on cross-checked, time- and location-matched evidence, with trust in meter readings scaled to accountability.
- Sound economics: fixed emissions, within-year green-budget reallocation without new minting, and explicit budgeting of anchoring costs.

8. Exemplary Claims

The following exemplary claims are provided to illustrate the subject matter of this provisional application and are not intended to limit it.

1. A computer-implemented method for decentralized processing of compute jobs, comprising: receiving a job specification at a scheduler of a ledger-anchored network; partitioning the job into micro-tasks; routing the micro-tasks to compute nodes; receiving, from each compute node, a signed compute receipt comprising a commitment to an output of a micro-task; validating the outputs using at least one of (i) redundant execution with comparison of outputs, (ii) random sampling and re-execution of segments of the computation, and (iii) trusted-execution attestation; computing a reward for each node using a multiplier derived from verified energy-source and efficiency data; releasing rewards upon successful validation; and recording hashes of the receipts and payouts on a ledger.

2. The method of claim 1, wherein each compute node is assigned an accountability level determined from one or more of verified operator identity, a signed agreement including audit rights, physical control of the node hardware, and independent facility certification; wherein a frequency or depth of the validating, and a weight given to trusted-execution attestation, are set according to the accountability level; and wherein a non-zero rate of validation is maintained at every accountability level.
3. The method of claim 2, wherein jobs classified as involving sensitive data are routed only to compute nodes whose accountability level meets a threshold and whose hardware supports confidential computing.
4. The method of claim 1, wherein the validating comprises: receiving from each of a plurality of nodes executing the same micro-task a commitment to a salted hash of that node's output; withholding revelation of any output until commitments from all of the plurality of nodes are recorded; and thereafter comparing the outputs.
5. The method of claim 4, wherein each job class is associated with one of an exact-match comparison mode, in which the revealed outputs pass if their hashes are identical, and a tolerance comparison mode, in which the outputs are revealed privately to assigned verifiers, each output is checked against its own commitment, and the outputs pass if they agree within a declared tolerance.
6. The method of claim 1, wherein the validating comprises: receiving from a node a commitment to a Merkle root computed over hashes of intermediate checkpoints of a computation; after receiving the commitment, randomly selecting checkpoints; requiring the node to open each selected checkpoint with an inclusion proof to the committed Merkle root; and re-executing segments corresponding to the selected checkpoints, wherein a number of selected checkpoints depends on an accountability level of the node.
7. The method of claim 1, further comprising selecting verifiers by stake-weighted random selection from a pool excluding any node that shares an operator, stake source or network cluster with the node being verified, wherein any environmental-score adjustment to verifier selection is capped.
8. The method of claim 1, wherein the network comprises validators that order and finalize entries on the ledger, the validators being selected by stake, and wherein the environmental score is used for routing of compute jobs and computation of rewards but not for selection of the validators.
9. The method of claim 1, wherein each compute receipt is serialized in a canonical encoding, comprises a signature over content including an identifier of the ledger and of a receiving contract, and is accepted only if a signer recovered or verified from the signature matches the node registered for the micro-task.
10. The method of claim 1, further comprising folding a plurality of compute receipts into a batch Merkle tree in which leaves and internal nodes are hashed with distinct prefixes, recording only a root of the batch Merkle tree on the ledger, and delivering to both the submitter and the node an inclusion proof for the respective receipt at the time of recording.
11. The method of claim 10, wherein a batch interval for recording roots differs according to a funding lane of the jobs whose receipts are included.
12. The method of claim 1, wherein the energy-source data comprise readings signed by a secure element of a metering device distinct from any compute enclave, a weight of the readings

being scaled according to an accountability level of a party physically controlling the metering device, and the readings being cross-checked against grid data and a reported compute load of the node.

13. The method of claim 1, wherein the multiplier is a VerdantScore computed from time- and location-matched energy proofs, efficiency metrics and verified sustainability behaviors, wherein certificate-only claims lacking time and location match are rejected, and wherein tier upgrades take effect only after a cool-down period, subject to audit and clawback.
14. The method of claim 1, further comprising a challenge window during which any participant may post a bond to challenge a result by re-executing designated samples against committed roots, a successful challenge resulting in slashing, re-adjudication and payment of a bounty, wherein a length of the challenge window is set per job class.
15. The method of claim 1, wherein rewards are computed as $\text{base_job_value} \times \text{node_type_multiplier} \times \text{tier_multiplier} \times \text{product}(\text{modifier_bonuses})$, and wherein node type is established by benchmark testing at onboarding and subsequent spot-checks.
16. The method of claim 1, wherein emissions follow a multi-year declining schedule within a fixed supply, and a green multiplier budget is capped within each year and burns if unused, such that environmental bonuses reallocate within an annual emission cap without increasing total supply.
17. The method of claim 1, wherein the scheduler applies three funding lanes comprising a protocol-funded default lane in which the submitter need not hold a wallet, an enterprise staked-priority lane providing queue priority without direct node payment, and an optional direct-pay bounty lane.
18. The method of claim 1, further comprising permitting a submitter whose job is not scheduled within a defined period to submit the job directly to a ledger contract that obliges scheduling of the job.
19. The method of claim 1, wherein each node operator holds a receipt-signing key authorized only to sign compute receipts and a separate key controlling earnings and stake, and wherein treasury funds are controlled by an m-of-n arrangement of independent keyholders.
20. A system comprising one or more processors, memory and program instructions operable to perform the method of any of claims 1 to 19, the system comprising a ledger interface, a scheduler, a node registry recording role, type, environmental tier and accountability level, a validation engine, a verifier selector, an energy-oracle and metering interface, a reward calculator, and storage adapters for on-ledger hashes and off-ledger encrypted artifacts.
21. A computer-implemented method of verifying computation performed by untrusted nodes, comprising: assigning to each node an accountability level based on identity, contractual, physical-control and certification criteria; setting, for each node, a verification intensity and a weight given to hardware attestation as a function of the accountability level, the verification intensity being greater than zero at every level; verifying outputs of the node at the set intensity; and releasing payment to the node only upon successful verification.

9. Definitions

- **Accountability level:** A classification of a node reflecting how accountable the party physically controlling its hardware is, based on identity verification, signed agreement and

audit rights, physical control of the hardware, independent certification and stake; used to set verification intensity, attestation weight and sensitive-data eligibility.

- **Commitment:** A hash of data (optionally combined with a random salt) published before the data are revealed, so that the data cannot later be changed without detection.
- **Compute receipt:** A structured record binding a micro-task's inputs, code, outputs, runtime metadata and checkpoint and attestation hashes, signed by the node and, after verification, by verifiers.
- **Comparison mode:** The rule for comparing redundant outputs for a job class: exact-match (identical hashes) or tolerance (agreement within a declared margin).
- **Spot-replay sampling:** Re-execution of randomly selected segments of a computation, selected after the node has committed to its checkpoints, to verify correctness probabilistically.
- **Inclusion proof (Merkle proof):** The sibling hashes along the path from a leaf to the root of a Merkle tree, allowing anyone to confirm that the leaf is included under a published root.
- **Forced inclusion:** A mechanism allowing a submitter to place a job directly on the ledger so that it must be scheduled.
- **VerdantScore:** An environmental score derived from verified energy sourcing, efficiency telemetry and sustainability behaviors, mapped to reward multipliers, routing and privileges.
- **Green budget:** A within-year allocation of emissions dedicated to environmentally weighted rewards; burns if unused; no net supply increase.
- **Node:** A participant device that contributes computing power to process jobs, verify results or support system functions, described by role, type, VerdantScore tier and accountability level.
- **EcoPrime Pools:** Groups of nodes with verified high VerdantScore tiers prioritized for sustainability-tagged jobs and paired with staking pools funded from the green budget.
- **EcoRoyalties:** Distributions to high-VerdantScore nodes earned per sustainability-tagged job or from recurring network-level allocations funded by transaction fees or the green budget.

- EcoChallenges: Periodic sustainability competitions open to all participants regardless of node type, with rewards such as tokens, badges or grants.
- EcoMatch: A partnership framework connecting research institutions, nonprofits and enterprises with sustainable compute pools.
- EcoBonus: A one-time boost layered on VerdantScore rewards for exceptional actions.
- The REVO Accord; SmartVote: The network's decentralized governance system; SmartVote requires a participant to pass a comprehension quiz before voting.
- Trusted Execution Environment (TEE): An isolated area of a processor in which code and data are protected from the rest of the system and which can produce attestation reports.
- Energy oracle: An external data feed providing time- and location-matched information about grid carbon intensity, renewable sourcing or utility certification.
- API; SDK: An application programming interface allowing outside software to submit jobs automatically; a software development kit of tools for integrating with the API.

10. Implementation Notes

- On-ledger footprint is minimized by recording batch roots of receipts; full artifacts are stored off-ledger.
- The scheduler respects sensitivity classes and accountability thresholds; sanctions and export filters run before admission.
- Randomness for spot-check selection and verifier selection is drawn from a source the node cannot predict or influence at commitment time.
- Stake, sample counts and challenge-window lengths are tuned so that the expected cost of cheating exceeds its expected gain at every accountability level.

11. Figures (ASCII)

FIG. 12 – Reward computation

```
payout = base_job_value x node_type_mult
```

```

        x tier_mult x PRODUCT(modifiers)
node_type_mult: General 1.0 | GPU 1.25
  | Verified Research 1.5 | Enterprise 1.75-2.0
tier_mult (VerdantScore Tier 0-4): 1.0 ... 2.0

```

FIG. 13 – Data trust and job safety

```

[Submit] -> [Static scan] -> [Policy classifier]
  -> [Sanctions/export] -> [Sandbox] -> Queue
Sensitive (PHI/PII) -> accountability Level 2+
  with confidential computing only
Ledger: hashes and receipts only

```

FIG. 14 – Storage topology

```

On-ledger: batch roots of receipts | payouts
  | stake | votes | node registry
IPFS / Arweave: immutable public records
  (PoP logs, governance votes)
Encrypted storage: large or sensitive artifacts
Each party: its own receipt + inclusion proof

```

FIG. 15 – Governance hardening

```

[Proposal] --(quiz gate)--> [Vote]
  +--> [Delegation] [Time-locks]
  +--> [Emergency pause <= 72h
    + public report + ratification]

```

FIG. 16 – Annual emissions cap and green budget

```

Year N emission cap = node rewards + green budget
Green budget -> environmental multipliers,
  EcoRoyalties, EcoPrime, EcoChallenges
Unused green budget at year end
  -> burned (no new minting)

```