Compute Substrate

A Permissionless Computation Layer Without Authority

Author: csd

2026-01-25

Abstract:
We present Compute Substrate, a permissionless proof-of-work network for producing and persisting speculative computation without semantic or outcome-level authority. Compute Substrate introduces a new class of infrastructure: a public cognition layer that is cryptographically persistent but structurally incapable of exercising power. Participants submit proposals (opaque computational outputs) and attestations (support signals) at a cost. The network aggregates these signals deterministically and commits them to a public ledger. Compute Substrate does not determine correctness, resolve outcomes, or trigger actions. Its sole purpose is to maintain a reproducible, cost-bearing history of what was proposed and supported over time, while remaining safe to ignore.

1. Introduction

Modern distributed systems increasingly rely on large-scale computation: predictions, rankings, simulations, and analyses generated by many independent actors. In practice, the outputs of these computations are often treated as decisions, triggering actions or reallocating resources. In these systems, temporal progression itself implicitly functions as a source of authority: actions occur because time advanced. Compute Substrate explicitly rejects this assumption. In this system, time may order events, but it can never legitimize action. This collapse of layers, allowing computation to accumulate authority, introduces fragility. Errors become catastrophic, incentives distort behavior, and systems attract adversarial pressure proportional to the consequences of being “right”. This paper explores a narrower question: can computation be scaled independently of authority? That is, can a system persist the results of computation without implying correctness, execution, or obligation?

Historically, computation has always been coupled to consequence: models guide decisions, predictions trigger actions, and outputs implicitly carry authority. Compute Substrate is the first distributed system that explicitly forbids this coupling at the protocol level. It introduces a new category of infrastructure in which cognition may scale arbitrarily while remaining structurally incapable of affecting the world.

We present Compute Substrate as a minimal affirmative answer. It is a network that computes and records speculation only. Authority, interpretation, and action are explicitly external. The system remains complete even if all recorded outputs are wrong or ignored forever.

Accordingly, Compute Substrate is defined as a substrate rather than an outcome-producing system. It does not resolve uncertainty or produce outcomes. Instead, it provides a neutral, persistent layer for recording computation under fixed cryptographic rules. The substrate is indifferent to meaning, correctness, and use. All interpretation, validation, and action occur outside the protocol. Informally, Compute Substrate can be understood as a system that extracts the informational output of intelligence while stripping away its ability to act.

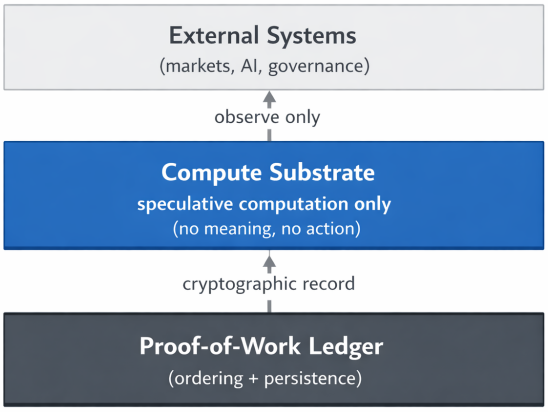

Figure 1. Layer separation in Compute Substrate.
The substrate records speculative computation without meaning or execution. All authority and action exist strictly outside the protocol.

2. Design Goals

Compute Substrate is designed around the following goals:

- Permissionless participation: anyone may join, propose, attest, or mine.
- No authority: the network never decides correctness or resolves outcomes.
- No execution: recorded outputs never trigger actions.
- Cost-bearing computation: contributing signals requires paying fees.
- Deterministic aggregation: all nodes derive identical state from the same history.
- Verifiable history: all state is reproducible from genesis.
- Safe to ignore: external systems may observe or disregard outputs without consequence.

These goals intentionally exclude many features common to “intelligence” or “governance” systems. The exclusions are essential to safety.

3. System Overview

3.1 Network Model

Compute Substrate is a peer-to-peer network secured by proof-of-work. Nodes perform three roles:

- **Full nodes:** validate blocks, maintain state, and serve data.
- **Miners:** construct blocks and perform proof-of-work.
- **Clients:** submit transactions and query aggregated outputs.

All roles are permissionless.

3.2 Blocks and Consensus

Blocks are linked by hash pointers and ordered by cumulative chainwork. The consensus rule is simple: The canonical chain is the valid chain with the highest total chainwork. Proof-of-work provides Sybil resistance and ensures that history is costly to rewrite. Block rewards and transaction fees incentivize inclusion and liveness, not correctness of computation. In Compute Substrate, proof-of-work secures temporal provenance rather than consensus truth: it makes public memory costly to forge, not authoritative to follow.

Unlike conventional blockchains, which use proof-of-work or proof-of-stake to secure economic state or execution outcomes, Compute Substrate uses proof-of-work solely to secure the cost of writing into a shared public memory. The network does not enforce correctness, resolve disputes, or assign authority to any computational result. It only ensures that inserting data into the global record requires irreversible external cost. In this sense, Compute Substrate secures temporal provenance rather than consensus truth.

3.2.1 Difficulty and Memory Production Rate

Proof-of-work secures temporal ordering by making history costly to rewrite. Difficulty primarily regulates the rate at which this public record grows, and it bounds the cost of reorganizing recent history. Compute Substrate does not attach semantic authority to on-chain artifacts, so manipulation of ordering yields limited payoff beyond inclusion and timing. Difficulty adjustment is therefore treated as a throughput and stability mechanism for the public record, not as a truth mechanism for computation.

3.2.2 Consensus and Authority

In most distributed systems, consensus is used as a decision mechanism: once a network converges on a state, that state carries immediate consequences. Assets are reassigned, actions are executed, or obligations are enforced. In such systems, consensus functions as de facto authority. Although protocols do not claim epistemic correctness, the presence of consequence causes participants and external observers to treat accepted state as if it were authoritative.

Compute Substrate rejects this coupling. In this system, consensus guarantees only reproducible agreement on history, not correctness, validity, or entitlement. The protocol assigns no rights, powers, or obligations to any recorded artifact. Inclusion in the canonical chain confers no semantic authority and produces no outcomes. By removing all protocol-level consequences from consensus, Compute Substrate ensures that agreement remains purely coordinative. Consensus establishes what was recorded and when, but never what is correct, valuable, or actionable. Authority, interpretation, and execution are strictly external to the protocol.

Compute Substrate is structurally inert: no on-chain artifact can cause side effects beyond its own reproduction in future state. This ensures that even perfect manipulation of the system cannot translate into power, only into additional records. The protocol cannot express commands, obligations, or triggers. All state transitions are internal to the ledger itself and carry no semantic force outside the system.This design enforces a structural separation between coordination and

consequence. Whereas most systems use consensus to decide, Compute Substrate uses consensus only to remember. This removes authority from the protocol layer entirely, such that the system is structurally incapable of deciding outcomes or conferring power.

3.2.3 Temporal Impotence

In conventional distributed systems, temporal ordering implicitly confers authority: once an event is included in the canonical sequence, it produces consequences. Compute Substrate enforces the opposite invariant. Although events are ordered in time, the protocol is structurally incapable of treating sequence as permission. No on-chain artifact can trigger execution, obligation, or decision. Time is preserved purely as memory, not as control.

3.3 Transactions

Transactions follow a UTXO model. Each transaction may include an optional application payload. Two payload types are defined:

- **PROPOSE:** submits a speculative computational output.
- **ATTEST:** expresses support for an existing proposal.

Transactions pay fees. Minimum fees are enforced for PROPOSE and ATTEST payloads to bound spam.

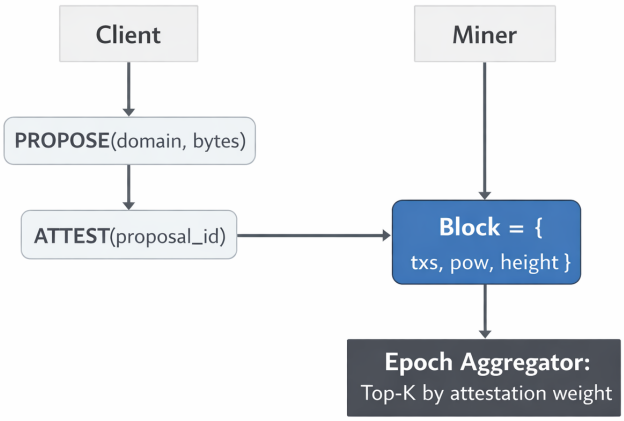

Figure 2. Transaction flow in Compute Substrate.
Clients submit speculative outputs (PROPOSE) and weight them (ATTEST). Miners include transactions in proof-of-work blocks. Epoch aggregation ranks proposals deterministically without implying correctness or triggering execution.

3.4 Proposals

A proposal consists of:

- a domain identifier (e.g. “finance”, “science”)
- an opaque payload (bytes)
- a timestamped inclusion height

The network does not interpret proposal contents. Meaning exists only to observers.

3.5 Attestations

An attestation references a specific proposal and contributes weight to it. Attestations are also opaque to the protocol; they simply increase a proposal's score within an epoch. Attestations contribute weight equal to the transaction fee paid (in base units).

3.6 Epochs and Aggregation

Time is divided into fixed-length epochs. Within each epoch, proposals are ranked by the sum of attestation weight they receive. Ties are broken deterministically by proposal identifier (lexicographic ascending). At the end of an epoch, the network derives a deterministic Top-K list for each domain. These lists are part of the canonical state and are queryable by any node. Importantly:

- Rankings do not imply correctness.
- Rankings do not trigger rewards or actions.
- Rankings may change across epochs or reorgs.

4. Incentives and Fees

Compute Substrate uses incentives only to ensure participation and liveness.

- Block rewards incentivize miners to secure the chain.
- Transaction fees incentivize inclusion and bound spam.

There are no rewards for being correct. Fees are paid regardless of outcome. This prevents computation from becoming authority-bearing.

5. State, Forks, and Reorganizations

Like any proof-of-work chain, Compute Substrate may experience temporary forks. Nodes track block headers, cumulative chainwork, and maintain undo logs for state transitions.

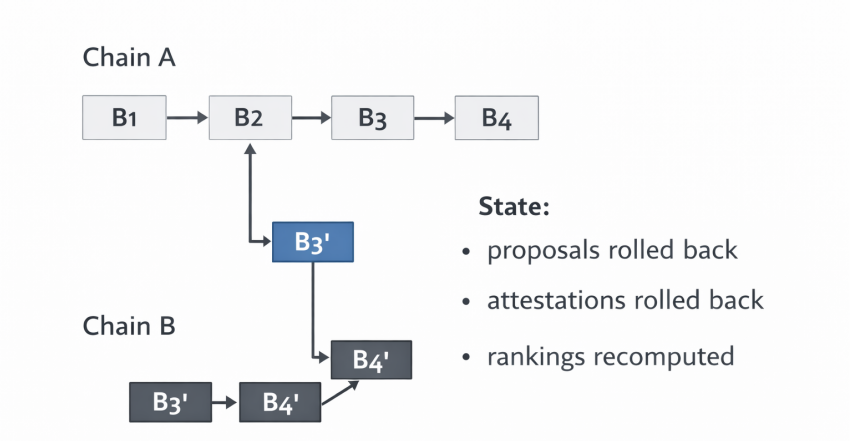

Figure 3. Forks and deterministic state rollback.
Competing chains may temporarily diverge. On reorganization, proposals and attestations are rolled back and rankings recomputed so that aggregated outputs always correspond to the canonical chain.

On reorganization:

- spent UTXOs are restored
- created UTXOs are removed
- proposal and attestation inserts are rolled back

This ensures that aggregated outputs always correspond to the canonical chain. Although the system does not execute actions, correctness of reorg handling is essential for determinism and verifiability.

6. Security Properties

6.1 Adversarial Model and Threat Surface

Compute Substrate does not secure assets, execution, or authority. Therefore adversarial strategies that seek to gain economic or control advantage by manipulating consensus yield no meaningful payoff. The only adversarial action available is the creation of additional records at personal cost, which does not degrade protocol determinism or confer influence. Adversaries may attempt to spam, censor, or reorder transactions, but such actions affect only inclusion and ordering, not the interpretation, correctness, or authority of recorded computation.

6.2 Adversarial Computation

Malicious or incorrect proposals are allowed by design. They are harmless because they carry no authority. Compute Substrate is not designed to be correct; it is designed to be reproducible.

6.3 Spam Resistance

Spam is bounded by transaction fees and proof-of-work costs.

6.4 No Oracle Attacks

Because outcomes are never resolved, there is no oracle to corrupt and no incentive to do so.

6.5 Miner Behavior

Miners may reorder or exclude transactions, but can only affect inclusion, not interpretation or execution.

6.6 Censorship

Many decentralized systems reduce centralized control while retaining semantic or outcome-level authority. Compute Substrate removes these layers entirely, leaving transaction inclusion as the only remaining censorship surface.

Because the protocol assigns no meaning, correctness, or authority to recorded computation, censorship can occur only at the level of transaction inclusion and cannot target interpretation, outcomes, or use. Once recorded, computational artifacts are reproduced deterministically and cannot be selectively removed without rewriting chain history. As in other proof-of-work systems, transaction inclusion may be delayed by individual miners, but no mechanism exists for semantic or retroactive censorship within the protocol.

7. Non-Goals

Compute Substrate explicitly does not attempt to:
- determine truth or correctness
- resolve real-world outcomes
- execute actions or trigger payments
- govern participants
- provide predictions with guarantees

- replace markets, oracles, or governance systems

These functions belong to higher layers. In particular, Compute Substrate is not designed to make better decisions, only to make it impossible for computation itself to decide.

8. Applications (External)

Compute Substrate is intended to be observed, not relied upon. External systems may use its outputs as:

- speculative inputs
- advisory signals
- planning data
- exploratory intelligence

Any authority to act on these outputs must exist entirely outside the protocol.

9. Conclusion

Compute Substrate demonstrates that temporal ordering can be separated from authority in a distributed ledger. By committing only speculative outputs under fixed cryptographic rules, the system enables public cognition without decision, control, or obligation. All interpretation, validation, and action are strictly external to the protocol, which remains complete even if ignored forever.