Republic: A Platform for the Orchestration of Compute through Cryptoeconomic Incentives

Republic Foundation

Abstract: Blockchain networks have allowed for the decentralized orchestration of compute to validate digital currencies and internet-based applications. This has resulted in economic networks of scale being developed entirely on the internet, with no coordinated physical presence. However, there is not yet a platform that directly leverages the decentralized coordination inherent in contemporary blockchain networks to provide generalized, high-performant compute. In this work, we propose a new economic network that is built solely to mediate the exchange of compute with a digital currency. Network validators are selected and coordinate new, valid blocks through a normalized, secure delegated proof of stake algorithm that takes into account both accumulated stake and the quality of compute the validator is providing, which is quantified via a new, discrete algorithm. The network's primary use-case will be to provide compute to participants in exchange for the underlying native token: thus, validators have multiple incentives to provide quality compute to network participants: economic incentives (from being selected for consensus and by being paid for their compute) and reputational incentives (by attracting more delegation from other participants). Instead of solely using compute as a method for cryptographic validation and execution of transactions or application-calls, we make it a central part of the platform, and allow it to function as a point of exchange for the underlying currency.

1 Introduction

Generalized, cryptographically verifiable distributed ledgers, in combination with peer to peer networks, have allowed for the proliferation of blockchain technology, which refers to any protocol or network that allows for the decentralized validation of arbitrary state transition functions, such as the transfer of a currency unit from one party to another or the issuance of a new collectible, without a centralized intermediary. Such protocols often leverage the aggregated compute of their participants to enforce consensus or scale the platform: in the case of Proof of Work platforms such as Bitcoin, the latest block (and canonical chain of past blocks) is determined by the participant who is the first to be able to compute a valid SHA-256 hash of the block header, added to a nonce, such as that it is less than some target value [1]. In alternative networks, such as Ethereum, a committee of validators are elected based on the amount of native currency they have locked, or staked, programmatically within the network, and use their computing power to validate new state transitions to the underlying ledger [2].

However, there does not yet exist a mainstream platform through which the underlying compute of participants can actually be used for use-cases other than the execution of native programs. For this, we need a way to incentivize validators, who we define as participants in the network who are

willing to validate and maintain their own ledger, to provide high quality compute to other participants. Validators who consistently provide usable and efficient compute to other participants earn incentives, both computationally via the network and via direct payments in the native currency from their peers. Programmatic incentives allow for those validators to have a higher probability of being in charge of proposing the next block, which determines the current canonical set of transactions at any time t.

2 Validation of Compute

Compute is primary method of exchange for the underlying currency, which we define as $REP. All validators who have a full node, or are currently participating as a peer in the protocol by actively relaying new messages, are eligible to provide their compute to other participants via the platform. We define this platform as a generalized exchange marketplace, where any participant with a valid amount of $REP is able to exchange their tokens for access to a certain amount of compute from a registered validator for a set period of time.

We thus define a generalized, blackbox validation function that takes as input a participating validator's hardware, and returns a quantitative value representing its quality. This rating is used two-fold: first, it has a direct weight on the validator's ability to be selected in future rounds of consensus, and second, its likelihood of being paid by other participants for its compute. This function, while running on the validator's machine, will be able to be cryptographically verified by others. This will ensure that every validator has a localized view and the same compute rating for other validators.

The compute of each validator is measured twice, first as an overall benchmark to determine the effectiveness of the compute it is providing to peers, and secondly as a generalized validation mechanism across the network to ensure that the validator is actually meeting the thresholds determined in its generalized benchmark. The generalized flow of a new compute transaction, which is a request of compute from one participant to a singular node, and the subsequent validation of the provider's compute, is visualized in 1.

Let F be a network-standard benchmark program (e.g. a transformer inference kernel), and let $H_F : B^* \to Z_p$ be an injective postcomposition function chosen by the network at genesis. For each benchmark round the network samples a public target $y \in \mathrm{Im}(H_F \circ F)$ and fixes a tolerance $\varepsilon_F > 0$. A GPU (or other accelerator) passes the benchmark iff it can discover a nonce n such that

$$\big|\, H_F\big(F(n)\big) - y \big| < \varepsilon_F.$$

The tuple $\langle n,\ F(n) \rangle$ is thereafter treated as a *performance commitment*: any future job must sustain throughput at (or above) the TFLOPs implied by $F(n)$ [3].

During the benchmark run, the validator records *check-points*— e.g. the activations after each transformer block—to generate a precise proof which demonstrates knowledge of all computations required to produce the desired result. The attestation package broadcast to the chain is then

$$\Pi = \big\{ n,\ \{a_i\}_{i=1}^{k} \big\},$$

where a_i are the checkpoint tensors. Peer validators verify:

(i) $\forall i :\ a_i = v_i$ for checkpoints v_i computed by the validator with input n.

(ii) $\big|\, H_F(F(n)) - y \big| < \varepsilon_F$.

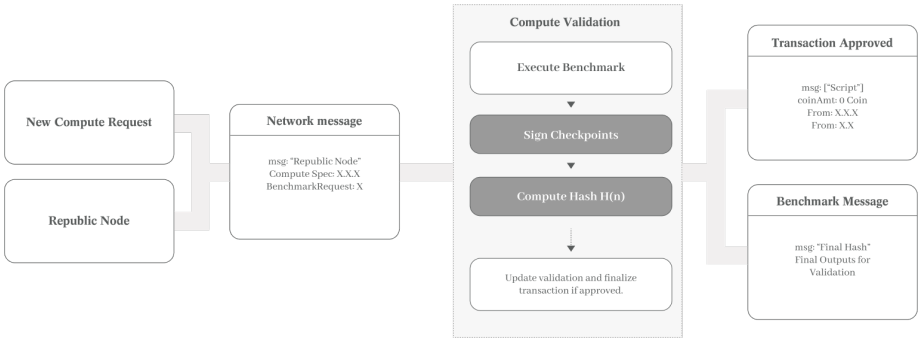

Figure 1: Validation of Compute

Let T_F be the TFLOPs/sec measured during the benchmark and T_J the TFLOPs/sec actually delivered for job J. Define the performance ratio $\rho = T_J/T_F$. Each validator maintains a scalar reputation score $R \in [0, 1]$. After job J we update

$$R \leftarrow R_{\text{new}} \;=\; \mathcal{R}\big(R_{\text{old}}, \rho\big),$$

where, by default,

$$\mathcal{R}(r, \rho) = \begin{cases} r + \alpha(1 - r)\rho & \text{if } \rho \geq 1 - \delta, \\ r\big(1 - \beta(1 - \rho)\big) & \text{otherwise,} \end{cases}$$

with protocol constants $\alpha, \beta, \delta \in (0, 1)$ controlling reward, penalty, and tolerance respectively [4]. Reputation R feeds directly into the committee selection weights (cf. Section 3) for the consensus protocol, tying a validator's ability to provide consistent compute with their likelihood of being able to verify the underlying currency.

This is a running calculation; every time a validator provides their compute to another participant, their compute rating is updated based on the specific usage incurred by the participant. Good experiences, ie, compute with low latency, uptime, and bandwidth, results in the compute rating increasing, while bad experiences will incur a penalty on the validator's compute rating.

An individual validator's compute rating can be verified by the entire network, just like their balance or stake. Thus, the compute rating becomes an independent part of the network over time.

3 Consensus

We define consensus as a distributed protocol through which participating validators can come to agreement on the canonical state of the blockchain. Republic leverages a standardized proof of stake (POS) protocol for its consensus algorithm, allowing for high throughput and scalability, while maintaining a low barrier to entry for any participant. Our POS implementation is a generalized implementation of the protocol implemented by Ethereum [2]. We make small alterations for how validators are selected at any given set of time by including a new weight, a quality of compute rating, for each validator associated interested in being selected to the committee. We also allow for delegation of stake from one validator to the other: this can be thought of as a native implementation of staking pools, and is an extension of the delegated stake mechanism described by [5]. We define

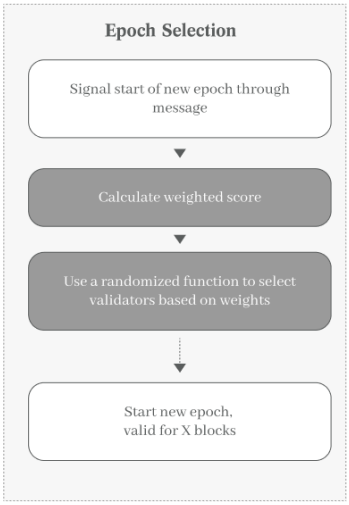

Figure 2: Epoch Selection

an epoch e as an unit of time, occurring over a certain number of blocks, which are simply data structures representing multiple transactions. At the beginning of each e, a committee c is selected from the set of all available validators, based on a weighted, randomized algorithm that takes into account the total number of tokens the validator has staked, the total number of tokens that have been delegated to the validator, and its compute-quality rating. These variables are formed into a score that is assigned to the validator at the beginning of e, and used by the algorithm to select c. c remains static for the duration of e, with the exception of removal and replacement due to misbehavior or inactivity. Before e formally starts proposing new blocks, all members in c exchange messages to sync on the current canonical chain, which is always assumed to to be the current longest chain. For each block b within e, a leader is selected from c. This leader is in charge of assembling the block from the current set of pending transactions as aggregated by the current set of validators, and proposes a new b for the chain. The leader then cryptographically signs b, and sends it to each validator within c. Each validator within c can then signal their approval for b by signing it, or rejecting it. By default, an invalid b which contains invalid transactions, does not have a proper signature, or is too large, will be rejected by all validators. Once at least $2/3$ of the stake accumulated within c approves b, it is then appended to the current canonical chain, and the c resumes the process for block $b + 1$. Stake accumulation refers to the weighing each validator's approval with their stake.

All validators are eligible to be part of the committee c of validators that will be in charge of proposing new blocks. In order to participate, they must sign a participation transaction (with 0 value) that indicates that they will be participating in subsequent consensus. This allows them to be part of the superset of validators that will be selected for consensus.

All participants who have at least 1 unit of currency can participate in consensus, even without the necessary hardware, through delegation. Delegation allows a participant to delegate their tokens to another validator for the duration of an epoch. These delegations, as described above, are used directly as a weight in the randomized selection process for a new committee. At the conclusion of any e, a participant can unstake their delegation, and restake to someone else.

Rewards are given individually to selected committee members after the conclusion of each epoch e, in proportion to their overall weighted score, as determined during the randomized committee selection process. A validator can set an optional, variable reward threshold percentage that will be distributed proportionally to their delegates: this creates a feedback loop, as validators who are more likely to distribute their rewards by rewarding other delegates will be more likely to receive further delegation in the future, creating an incentive for net decentralization.

Once a block b has been appended to the blockchain, it is simply an optimistic commitment:b is finalized at a block checkpoint t, which is simply a future block that is a direct descendant of b. In order to achieve this, we use a generalized finality gadget derived directly from Parity Lab's GRANDPA implementation [6]. Finality is dependent on the entire superset of validators, and is independent of the committee c chosen to propose an individual block for the blockchain. We define a checkpoint block, b_c, which refers to a certain number of blocks before it through successive references. These checkpoint blocks are predefined, and are set to occur at a specific point in time. At b_c, all validators are given the opportunity to vote with their stake on the current canonical chain by attesting that this latest block is valid. Once b_c is approved, all blocks preceeding it are finalized.

4 Reputation-Based Slashing

We enforce slashing as a cryptoeconomic primitive to ensure that validators do not misbehave, either during consensus or while providing compute to their peers.

Slashing is not triggered solely by cryptographic faults (e.g. double-signing) but also by a validator's *long-run behavioural record*. The protocol therefore couples the quantitative reputation score $R \in [0, 1]$ with a stake-cutting mechanism that progressively removes economic weight from persistently under-performing or malicious actors. A slash event is emitted whenever *any* of the following occurs:

(a) **Consensus Faults.** Double proposal, equivocation, invalid block content, or failure to vote within the epoch timeout.

(b) **Compute Misconduct.** Performance ratio $\rho = T_J/T_F$ drops below some minimum acceptable performance ratio $\delta_{\min}$ for m consecutive jobs.

(c) **Reputation Degradation.** Reputation falls below the lower threshold $R < R_{\text{crit}}$.

(d) **Delegated Collusion.** A delegator whose aggregate vote-weight is $> w_{\max}\,\%$ concentrated in already-slashed validators.

Thus, reputational-based slashing is not only based on cryptoeconomic incentive, but also long-term reputational harm: a validator that was slashed will be less likely to draw delegates, and will be less likely to have their compute used by their peers, therefore making slashing a generalized social proof that has impacts both directly within the protocol and external to it.

5 Currency and Primary Use-Case

Republic operates under a standardized, account-based model, as described by Wood in the Ethereum Yellow Paper [7]. Each account maintains a public address, utilized by other accounts for payments,

a hidden public-private key pair for signing new transactions, a balance indicating the total amount of $REP it has, a scalar representing the total number of rewards it has received from participating in consensus, and a nonce indicating the total number of transactions it has sent. Validator accounts additionally maintain a dynamic compute rating.

A transaction on Republic has the following standard fields:

1. Amount of $REP being sent
2. Public Address of the recipient
3. Public Address of the sender
4. Message: An optional field dictating the type of transaction, payment or compute
5. Fee limit, representing the maximum fee for the transaction
6. The signature of the sender
7. Validation: Cryptographic validation of compute, only for compute transactions

Republic will have two types of transactions: normal, which is a simple payment transaction, and a compute transaction, which is a transaction allocating access to a certain amount of compute. A compute transaction indicates to the network that the reciepient is allocating their compute to the sender for a certain period of time, and that the quality of compute provided must be rated, utilizing the same methodology described in section 2.

The primary use-case of Republic will be providing a generalized, open-compute marketplace for its participants. All validators will have the opportunity to provide their compute, in exchange for a fixed rate of currency. This process will be directly integrated into the network, with compute ratings being dynamically validated by other nodes. This process is modeled as an implementation to the generalized algorithm described in section 2. Whenever a new compute transaction is initiated by a user, the validate function is run, providing an up-to-date representation and score for the quality of the validator's compute. A compute transaction is rejected if the cryptographic validation provided in the transaction is not valid.

6 Network

In blockchain technologies, a standardized peer to peer network is often used to relay messages between nodes. Such an implementation often involves two distinct architectures: a gossip protocol is used to propagate information, while a distributed hash table is used to coordinate

Nodes within the Republic network maintain dynamic whitelists and greylists to manage peer connections effectively. The whitelists comprise nodes that consistently demonstrate reliability and security, while greylists serve as backups or potential candidates awaiting further validation. Peer selection within the network utilizes deterministic pseudorandom algorithms, significantly enhancing security through randomized peer matching across communication rounds, reducing the possibility of targeted attacks. We assume that the gossip protocol is implemented alongside a pre-defined node discovery and storage protocol. This can be based on an existing implementation such as Kademila or Chord, or be a new protocol entirely

New node integration into the Republic network occurs smoothly and securely through designated seed nodes or trusted entry points. Seed nodes are operated by reputable, altruistic community participants, and they provide initial network metadata—including client versions, synchronization states, and cryptographic nonces—to rapidly onboard new nodes. Altruistic and economically incentivized rational nodes immediately synchronize new participants, ensuring prompt and accurate data propagation. The visualization of a respective sizes of a node's whitelist and greylist, as stored in a distributed hash table following a prespecified protocol, is visualized in 3.

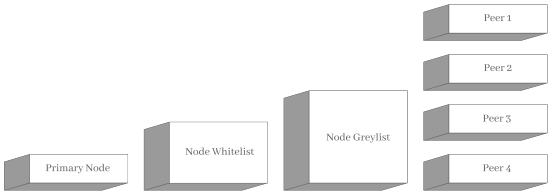

Figure 3: Peer to Peer Network Visualization

Republic also incorporates a sophisticated reputation management system combined with a Proof of Misbehavior (POM) protocol, offering robust and proactive security measures. The POM mechanism continuously evaluates node behavior based on accuracy, timeliness, and reliability of transmitted data. Nodes identified distributing incorrect, malicious, or otherwise harmful information accumulate negative reputation scores. When these scores surpass defined thresholds, nodes face demotion to greylist status or removal from active network participation, effectively isolating potential threats and preserving overall network integrity.

7 Conclusion

Republic is a generalized system optimized for the orchestration of compute without a centralized third party. Participants are able to leverage a high performance blockchain, powered by a custom delegated proof-of-stake mechanism, to get compute from the validators in exchange for the native token. A secure peer to peer network ensures that messages are able to be exchanged with low latency while maintaining security. Algorithmic validation of compute ensures that the quality of compute being provided by validators is directly tied to their ability to be chosen as a potential proposer for a future block, tying in both validation rewards and consensus to the ancillary compute exchange mechanism.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," May 2009. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf.

[2] V. Buterin, D. Hernandez, T. Kamphefner, *et al.*, *Combining ghost and casper*, 2020. arXiv: 2003.03052 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2003.03052.

[3] J. Smith and K. Lee, "Proof of execution for decentralised compute networks," OpenCompute Foundation, Tech. Rep., 2025.

[4] L. Zhang, *On chain reputation for performance-sensitive protocols*, Dyphira Research Note, 2025.

[5] BitShares. [Online]. Available: https://github.com/BitShares/bitshares/wiki/Delegated-Proof-of-Stake.

[6] A. Stewart and E. Kokoris-Kogia, *Grandpa: A byzantine finality gadget*, 2020. arXiv: 2007.01560 [cs.DC]. [Online]. Available: https://arxiv.org/abs/2007.01560.

[7] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.