Following on my work revolving around the Stateless AA proposal, I've raised an idea " [Proposing New Entity for Stateless Account abstraction- Trustless State provider Allowing TXs to enter Alt-mempool without holding the whole state

. I've studied @gballet 's brilliant article Verkle Tries Overview

and wanted to contribute to it.

The challenge at hand is to enable stateless verification of blocks in the blockchain system. The goal is to allow clients to verify the correctness of individual blocks without needing a full state. Instead, clients can rely on compact witness files generated by state-holding nodes, which contain the portion of the state accessed by the block along with proofs of correctness. This stateless verification offers several benefits, including supporting sharding setups and reducing reliance on validators.

To achieve this, my proposed solution focuses on adjusting gas costs for operations based on the witness size required for each operation, optimizing gas costs, and promoting efficient peer-to-peer interactions. The solution encompasses various key components and strategies to enhance the stateless verification process.

First of all, State-holding nodes play a critical role in witness generation, generating witnesses that encapsulate the zk proofs of the latest state correctness.

To enhance witness generation efficiency, I'd advocate the utilization of aggregated sub-vector commitment. This approach empowers state-holding nodes to update witnesses efficiently as the state evolves over time. The inclusion of zk-SNARK proofs in these witnesses further ensures compactness, reducing witness sizes and improving storage and data transmission efficiency.

The advantages of zk-SNARK-based witnesses are two-fold. First, they offer enhanced compactness due to their succinct nature, resulting in smaller witness sizes. This optimization significantly improves resource utilization and overall verification process efficiency. Second, zk-SNARK proofs provide improved privacy by enabling verification without revealing sensitive or confidential data. This aspect is especially crucial when handling state data that requires protection.

In terms of gas costs, the incorporation of zk-SNARK proofs in witnesses brings substantial reductions for specific operations. Account reads, encompassing operations like calls and EXTCODEHASH, benefit from efficient zk-SNARK proof verification, leading to lower gas fees. Additionally, witnessing SLOAD and SSTORE operations with zk-SNARK proofs effectively reduces gas costs, optimizing the consumption during state reads and writes. Moreover, the execution of smart contract code, which often involves accessing various parts of the state, can significantly minimize gas costs by employing witnesses with zk-SNARK proofs.

To implement this proposal, I'd plan to integrate aggregated sub-vector commitment into the witness generation process of state-holding nodes. Concurrently, we will develop an efficient mechanism to update zk-SNARK-based witnesses, ensuring synchronization with the latest state. Rigorous testing and auditing will be conducted to verify the correctness and security of this proposed witness generation mechanism.

Additionally, to ensure widespread availability, I suggest a sharded approach to witness generation and distribution. Each shard can have dedicated witness generators responsible for creating and publishing the necessary witness data, while a distributed network of nodes replicates and stores witnesses for improved accessibility.

subsequently, Handling cross-shard transactions efficiently is a fundamental requirement for a sharded system. The stateless light client proposal can include equipping the stateless Light Client to retrieve and validate witnesses from multiple shards involved in a cross-shard transaction, ensuring seamless interactions.

To optimize witness retrieval and minimize redundancy, a recommendation can be to implement a caching mechanism within the Light Client. This mechanism can store previously accessed witnesses, reducing the need for redundant witness requests, particularly for transactions accessing the same shard multiple times.

Furthermore, I still explore the use of zk-SNARKs (zero-knowledge proofs) for off-chain witness sharing to alleviate on-chain gas costs. Users can obtain zk-SNARK proofs from state-holding nodes off-chain, and only submit the succinct verification data or proof hash on-chain when necessary.

To maintain optimal efficiency, we propose a dynamic gas cost adjustment mechanism that continuously monitors network conditions and adjusts gas costs based on witness generation and validation demands. This adaptive approach ensures gas costs remain efficient and responsive to changing network dynamics.

To further optimize gas costs, an idea would be to introduce multi-level batch processing. Users can group transactions into batches, which can then be aggregated into super-batches for processing. This hierarchical batching approach significantly reduces per-transaction overhead, benefiting both on-chain and off-chain interactions.

A hybrid approach to witness sharing, combining on-chain and off-chain methods, can strike a balance between gas costs and efficiency. Critical parts of witnesses can be shared on-chain, while larger portions can be retrieved off-chain, optimizing overall gas consumption.