OpenST

A FRAMEWORK FOR BUILDING TOKEN ECONOMIES

Benjamin Bollen, Martin Schenck
for OpenST Foundation

Abstract

We present improvements to the OpenST protocol. OpenST is a framework powered by Ethereum to build token economies. We lay out in detail two contributions, OpenST Mosaic and OpenST Gateway, which work together to scale Ethereum. OpenST Mosaic is a layer-2

1. Introduction

2. OpenST

2.1. Use Cases

2.1.1 BT

2.1.2 DApp

3. Related Work

Verifiers' Dilemma [1]

Interblockchain Communication [2]

Casper FFG [3]

4. Our Contribution

5. Openly Scaling Blockchains

5.1. OpenST Gateway

OpenST Gateway enables chain-to-chain transfer of state objects.

5.2. OpenST Mosaic

OpenST Mosaic enables chain-to-chain state transfer. That allows us to secure an insecure chain's state on a secure chain. We call the secure chain *origin* and the insecure one *auxiliary*. In order to be blockchain proposal mechanism agnostic, we layer Casper [3] on top of both chains.

In order to transfer and verify state, it is required to transfer state bi-directional between auxiliary and origin. That way, auxiliary can verify that its state has been finalised on origin and the set of validators, that is tracked on origin, is also known to auxiliary.

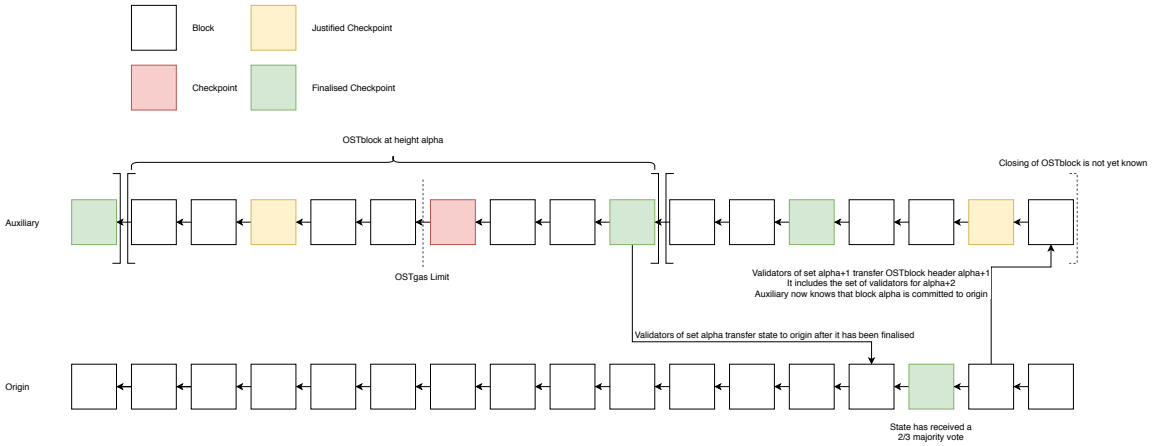

Figure 1: The Mosaic State Transfer

Validators are actors that vote on state. They are similar to validators in Casper [3]. Their deposit is stored on origin. Similarly to Casper, their deposit rises and falls with rewards and penalties, respectively. When we say "$\frac{2}{3}$ of validators", we equally refer to the deposit-weighted fraction. However, there are different kinds of voting message types. At this stage, we assume a fixed set of validators, $\mathcal{V}$. A dynamic set of validators will be described later.

Validators can vote on a number of events: (a) state transfers to auxiliary, (b) checkpoint justifications on auxiliary, and (c) state transfers to origin. When state is transferred from origin to auxiliary, the validators vote according to the Casper rules like the state transfer represents a checkpoint. On auxiliary, checkpoints are justified by supermajority links as described in Casper. Casper applied to auxiliary the same way its application to Ethereum is described in the Casper paper. When auxiliary's state is transferred to origin, a simple $\frac{2}{3}$ majority vote on that state commits it to origin.

OSTblocks are a compressed representation of auxiliary's transactions. Transactions in an OSTblock each represent one block of the auxiliary chain. An OSTblock transaction represents the state change from the beginning of its associated block to the end of that block. The OSTblock header of block $\mathcal{B}_\alpha$ at height α contains:

1. A link to the previous OSTblock
2. The signatures of the validators in $\mathcal{V}_\alpha$
3. The root hash of the state tree of auxiliary
4. The set of validators for the next OSTblock, $\mathcal{V}_{\alpha+1}$

We do not store the body of an OSTblock on origin, only the header. The content can be re-created at any time from the auxiliary chain and proven with the state root.

Process:

1. Open a new OSTblock $\mathcal{B}_\alpha$ at height α: copy state from origin to auxiliary
2. Auxiliary creates blocks
3. Close $\mathcal{B}_\alpha$: copy state from auxiliary to origin
4. Open $\mathcal{B}_{\alpha+1}$

In order to be block proposal mechanism agnostic, we layer Casper [3] on top of auxiliary. Validators vote on supermajority links. Auxiliary must follow Casper's fork choice rule. Checkpoints get finalised. The validators can transfer the state of a finalised auxiliary checkpoint to origin. A validator uses the Mosaic Gateway to transfer the state to origin. On origin, a $\frac{2}{3}$ majority vote is required by the validators in order to commit the state on origin. The state is regarded economically finalised when the block that contains the commit is considered economically final on origin.

On the auxiliary chain, *OSTgas* is the equivalent to gas on Ethereum. For transactions to be included in blocks on auxiliary, a fee is paid for the consumed OSTgas.

Validators could commit every finalised state from auxiliary to origin. However, that may be infeasible due to constraints on origin or from an economic viewpoint of the validators. Therefore, we propose points in the life of the auxiliary chain where the validators must close the OSTblock. Whenever a certain, fixed limit of OSTgas has been spent on auxiliary since the opening of the OSTblock, the validators must close the OSTblock at the next finalised auxiliary checkpoint.

Actors on the auxiliary chain have an economic incentive to finalise their state on origin. The block creators of auxiliary will pay a fraction of their earned OSTgas to the validators as a fee when the validators move the state to origin. Due to the OSTblocks, validators can only copy the state from auxiliary to origin after the state of origin has been copied to auxiliary. Thus, validators are incentivised to close an OSTblock and open a new one in order to earn the OSTgas fee for the transfer of auxiliary's state.

The set of validators needs to be able to change. The logic is similar to that of Casper with the notable difference that, instead of dynasties, we use OSTblocks. When a potential validator sends a *deposit message* on origin at OSTblock height α, then the validator ν will join the validator set at OSTblock height $\alpha + 2$. Analogous to Casper, we call $\alpha + 2$ the validator's *start dynasty*, $\mathrm{DS}(\nu)$.

6. Securing Simple User Experience

6.1. Token Holder Contracts

APIs; no need to follow the chain.

7. Platform

7.1. Token Rules

7.2. OpenST Platform

8. Outlook

8.1. Neo and Cardano

9. Conclusion

References

[1] Luu, L., Teutsch, J., Kulkarni, R. & Saxena, P. Demystifying incentives in the consensus computer. *IACR Cryptology ePrint Archive* **2015**, 702 (2015).

[2] Kwon, J. & Buchman, E. Bitcoin: A peer-to-peer electronic cash systems (2017). URL `https://cosmos.network/docs/resources/whitepaper.html`.

[3] Buterin, V. & Griffith, V. Cosmos: A network of distributed ledgers (2017). URL `https://github.com/ethereum/research/blob/master/papers/casper-basics/casper_basics.pdf`.