MapReduce for plasma-Blockchains

Christian Reitwießner
chris@ethereum.org

1 Introduction

The plasma system [JP17] defines a structure of interconnected blockchains arranged in a tree structure that promise scalable smart contracts. One of the key ideas there is that each of the blockchains regularly store their current block hash in their parent chain so that users can go to the parent chain and challenge potentially invalid state transitions in the child chain.

Here, the scalabality does not only come from the fact that blockchains are relieved from their load by creating a big number of smaller chains. Scalability is only achieved once a user does not have to verify every single transaction that is sent to the system. If, for example, a user only cares about a single smart contract that resides in a single chain that is a leaf of the system, it is sufficient for the user to verify this leaf and all nodes on the path to the root chain. If a transaction is commited by means of block hashes all the way up to the root chain and there is no invalid state transition in the chains on the way up to the root, the user can be reasonably sure that the transaction cannot be declared invalid by other users.

This system still does not solve the scalability problem: As long as the smart contract only "lives" inside a single blockchain, it can only process a limited amount of transactions. While this might be enough for some use-cases, a token contract can easily reach this limit.

The system would scale, if the token contract exists on all of the blockchains and it is possible to move tokens up and down the tree. Users would have accounts in only one or perhaps some of the chains and only watch the paths to the root from chose chains. In such a simple model, an attacker could just select a chain that is mostly unused, take it over, create an invalid state transition that creates tokens out of thin air and then move these tokens up to the root. If nobody is watching the attacked chain (or the attacker can turn off their computers or censor their transactions), the attacker is safe as soon as he or she is able to move the tokens far enough up.

Creating tokens out of thin air is a violation of the perceived invariants of a token contract and thus, such invariants should be checked in each of the chains: If we add a condition to the smart contract in each chain where the total balance held in direct child chains of this chain is recored, then an attacker can still create tokens in child chains, but he or she can only move tokens out of these flawed chains up to their total balance. For users that are not interested in these chains, the situation would not change: For them, someone took out tokens from a pool that exist in this pool, but it is not relevant who did it. In effect, the attacker of course steals tokens from users who that have accounts in the child chains, but at least the impact of the attack is confined to chains that are not properly watched.

In the next sections we will give examples of smart contract systems and how they can be distributed among a plasma system.

2 Assumptions

We assume that a user is watching a finite set of leaf chains and all chains on the path to the root from these leaf chains.

If the following two properties are invariants of a single blockchain, then they translate to the full blockchain if a user always only checks the path to the root.

1. if the balance of an account decreases, there has to be a valid signature by the owner
2. the total sum of all balances cannot change

Simple solution: Consider child chains as actual accounts.

References

[JP17] Vitalik Buterin Joseph Poon. Plasma: Scalable autonomous smart contracts. `http://plasma.io/plasma.pdf`, 2017.