EIP 1186: RPC-Method to get Merkle Proofs - eth_getProof (DRAFT)

Return to Ethereum ERCs

: **Note**: The following is an excerpt from the official Ethereum site. It is provided here as a connivence and is not authoritative. Refer to the original document as the authoritative reference.

Table 1: Data sheet for RPC-Method to get Merkle Proofs - eth_getProof

Title	RPC-Method to get Merkle Proofs - eth_getProof
Author	Simon Jentzsch, Christoph Jentzsch
Status	Draft
Created	2918-06-24
Description	http://eips.ethereum.org/EIPS/eip-1186
Specification	http://eips.ethereum.org/EIPS/eip-1186#Specification
Category	Interface

Simple Summary

One of the great features of Ethereum is the fact, that you can verify all data of the state. But in order to allow verification of accounts outside the client, we need an additional function delivering us the required proof. These proofs are important to secure Layer2-Technologies.

Abstract

Ethereum uses a Merkle Tree to store the state of accounts and their storage. This allows verification of each value by simply creating a Merkle Proof. But currently, the standard RPC-Interface does not give you access to these proofs. This EIP suggests an additional RPC-Method, which creates Merkle Proofs for Accounts and Storage Values.

Combined with a stateRoot (from the blockheader) it enables offline verification of any account or storage-value. This allows especially IOT-Devices or even mobile apps which are not able to run a light client to verify responses from an untrusted source only given a trusted blockhash.

Motivation

In order to create a MerkleProof access to the full state db is required. The current RPC-Methods allow an application to access single values (`eth_getBalance`, `eth_getTransactionCount`, `eth_getStorageAt`,`eth_getCode`), but it is impossible to read the data needed for a MerkleProof through the standard RPC-Interface. (There are implementations using leveldb and accessing the data via filesystems, but this can not be used for production systems since it requires the client to be stopped first - See https://github.com/zmitton/eth-proof)

Today MerkleProofs are already used internally. For example, the Light Client Protocol supports a function creating MerkleProof, which is used in order to verify the requested account or storage-data.

Offering these already existing function through the RPC-Interface as well would enable Applications to store and send these proofs to devices which are not directly connected to the p2p-network and still are able to verify the data. This could be used to verify data in mobile applications or IOT-devices, which are currently only using a remote client.