davFan Documentation

Release 0.0

TimberHouse

May 13, 2019

Contents

CHAPTER 1

BitCoin

1.1 Blockchain

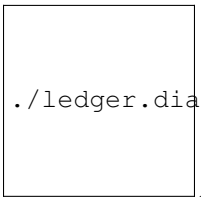
Let's now turn our attention from the world state to the blockchain. Whereas the world state contains a set of facts relating to the current state of a set of business objects, the blockchain is an historical record of the facts about how these objects arrived at their current states. The blockchain has recorded every previous version of each ledger state and how it has been changed.

The blockchain is structured as sequential log of interlinked blocks, where each block contains a sequence of transactions, each transaction representing a query or update to the world state. The exact mechanism by which transactions are ordered is discussed elsewhere; what's important is that block sequencing, as well as transaction sequencing within blocks, is established when blocks are first created by a Hyperledger Fabric component called the **ordering service**.

Each block's header includes a hash of the block's transactions, as well a copy of the hash of the prior block's header. In this way, all transactions on the ledger are sequenced and cryptographically linked together. This hashing and linking makes the ledger data very secure. Even if one node hosting the ledger was tampered with, it would not be able to convince all the other nodes that it has the 'correct' blockchain because the ledger is distributed throughout a network of independent nodes.

The blockchain is always implemented as a file, in contrast to the world state, which uses a database. This is a sensible design choice as the blockchain data structure is heavily biased towards a very small set of simple operations. Appending to the end of the blockchain is the primary operation, and query is currently a relatively infrequent operation.

Let's have a look at the structure of a blockchain in a little more detail.



A blockchain B containing blocks B0, B1, B2, B3. B0 is the first block in the blockchain, the genesis block.

In the above diagram, we can see that **block** B2 has a **block data** D2 which contains all its transactions: T5, T6, T7.

Most importantly, B2 has a **block header** H2, which contains a cryptographic **hash** of all the transactions in D2 as well as with the equivalent hash from the previous block B1. In this way, blocks are inextricably and immutably linked to each other, which the term **blockchain** so neatly captures!

Finally, as you can see in the diagram, the first block in the blockchain is called the **genesis block**. It's the starting point for the ledger, though it does not contain any user transactions. Instead, it contains a configuration transaction containing the initial state of the network channel (not shown). We discuss the genesis block in more detail when we discuss the blockchain network and channels in the documentation.

1.2 Blocks

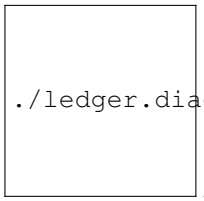
Let's have a closer look at the structure of a block. It consists of three sections

- **Block Header**

  This section comprises three fields, written when a block is created.

  - **Block number**: An integer starting at 0 (the genesis block), and increased by 1 for every new block appended to the blockchain.
  - **Current Block Hash**: The hash of all the transactions contained in the current block.
  - **Previous Block Hash**: A copy of the hash from the previous block in the blockchain.

  These fields are internally derived by cryptographically hashing the block data. They ensure that each and every block is inextricably linked to its neighbour, leading to an immutable ledger.

  Block header details. The header H2 of block B2 consists of block number 2, the hash CH2 of the current block data D2, and a copy of a hash PH1 from the previous block, block number 1.

- **Block Data**

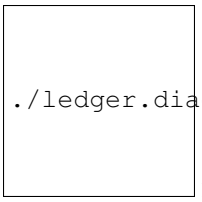
  This section contains a list of transactions arranged in order. It is written when the block is created by the ordering service. These transactions have a rich but straightforward structure, which we describe *later* in this topic.

- **Block Metadata**

  This section contains the time when the block was written, as well as the certificate, public key and signature of the block writer. Subsequently, the block committer also adds a valid/invalid indicator for every transaction, though this information is not included in the hash, as that is created when the block is created.

1.3 Transactions

As we've seen, a transaction captures changes to the world state. Let's have a look at the detailed **blockdata** structure which contains the transactions in a block.



Transaction details. Transaction T4 in blockdata D1 of block B1 consists of transaction header, H4, a transaction signature, S4, a transaction proposal P4, a transaction response, R4, and a list of endorsements, E4.

In the above example, we can see the following fields:

- **Header**

  This section, illustrated by H4, captures some essential metadata about the transaction – for example, the name of the relevant chaincode, and its version.

- **Signature**

  This section, illustrated by S4, contains a cryptographic signature, created by the client application. This field is used to check that the transaction details have not been tampered with, as it requires the application's private key to generate it.

- **Proposal**

  This field, illustrated by P4, encodes the input parameters supplied by an application to the smart contract which creates the proposed ledger update. When the smart contract runs, this proposal provides a set of input parameters, which, in combination with the current world state, determines the new world state.

- **Response**

  This section, illustrated by R4, captures the before and after values of the world state, as a **Read Write set** (RW-set). It's the output of a smart contract, and if the transaction is successfully validated, it will be applied to the ledger to update the world state.

- **Endorsements**

  As shown in E4, this is a list of signed transaction responses from each required organization sufficient to satisfy the endorsement policy. You'll notice that, whereas only one transaction response is included in the transaction, there are multiple endorsements. That's because each endorsement effectively encodes its organization's particular transaction response – meaning that there's no need to include any transaction response that doesn't match sufficient endorsements as it will be rejected as invalid, and not update the world state.

That concludes the major fields of the transaction – there are others, but these are the essential ones that you need to understand to have a solid understanding of the ledger data structure.

CHAPTER 2

Indices and tables

- search

2.1 License

The project is licesed under the BSD license.