Proof of existence & proof of origin on the blockchain

Robbert Jan van der Pas

12zUyQADZaPwxYiipVjMgrwEyzkaWgeCev

Author Note

Creating a mathematically proven audit trail for documents, transactions, receipts backed by a public or private blockchain.

Table of Contents

Introduction

Blockchain technology gives us the tools to create decentralized solutions that allow for mathematically proven proof of existence & proof of origin. Imagine this:

You have a degree from the University of Amsterdam, however when you go to work in another country you might have to go trough the process of proving that degree is real.
What if you can prove this mathematically that would save time & money.

This paper describes how a solution can be implemented that allows for proof of existence & proof origin. So in the example of the University of Amsterdam, you'd be able to prove that the degree existed from a certain date onwards and that it was issued by the University of Amsterdam.

Proof of Existence

Proof of Existence means that you can prove, irrefutably, that a certain document, transaction, receipt and so on existed at a certain point in time.

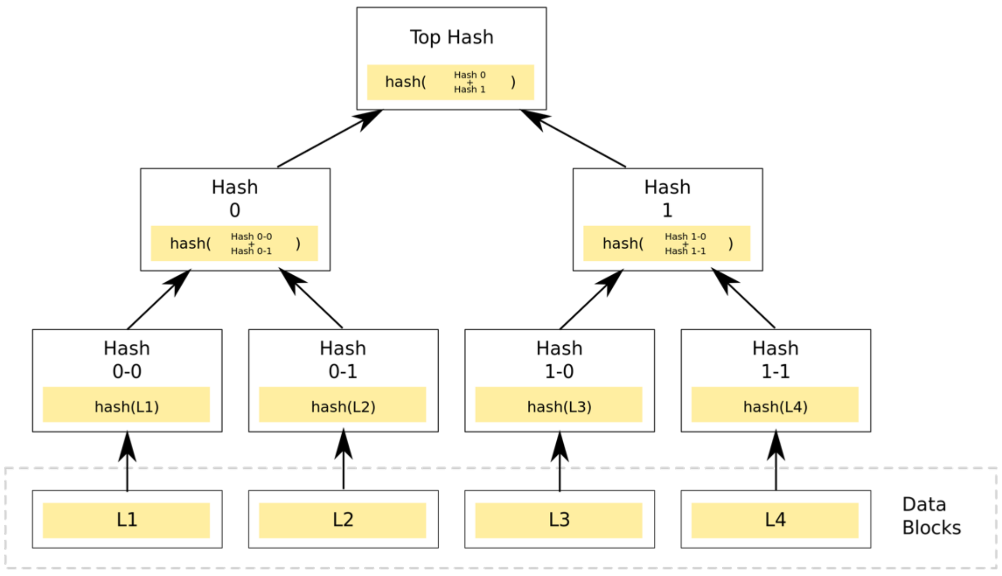
There are several ways of doing this, but the most common is to utilize a SHA-256 hashing function to generate a unique hash of the source data.

However, at the time of writing Bitcoin can handle handle approximately five transactions per second and each transaction costs approximately $0.10 USD. Therefore, this solution would not be scalable, when you're processing thousands of transactions per day.

To be able to record proof of existence in a way that is scalable we'll need to look into other methods.

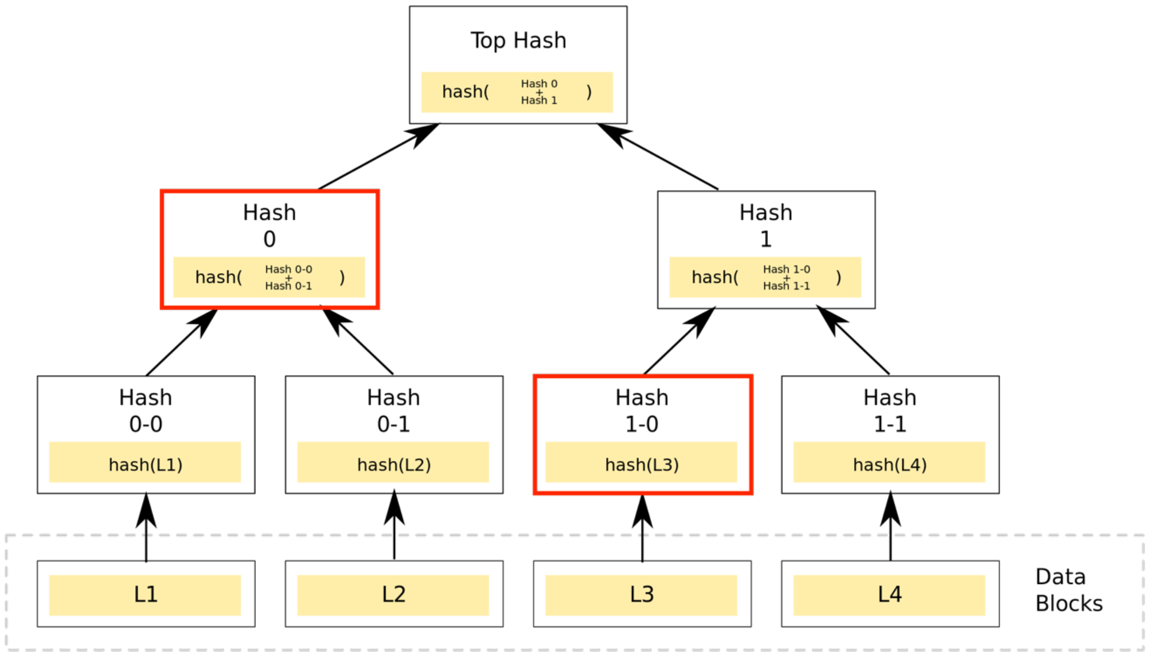
Let's say, we have a collection of 4 contracts and we wanted to store proof of existence on the blockchain. We start out by hashing each contract by utilizing a SHA-256 hash function.

Followed by collating these hashes into a Merkle Tree.



Then, by publishing the Merkle Root on the blockchain in a single transaction, we can create a receipt for all 4 contracts containing everything needed that a certain document existed at a certain point in time. For example, to proof that document L4 existed at the certain point of time, I need to record the following data onto the receipt:

- Blockchain transaction Id (which contains the merlke root)
- Hash 0
- Hash 1-0



By re-calculating the hash of my original document L4, combining it with the hash of L3 we should get hash 1. By combining hash 1 & hash 0 we should get the Merkle root recorded in the blockchain transaction. If not, the document is not the same as recorded when generating the Merkle tree.

Proof of Origin

For some use cases, proof of existence is not enough. In case of the University of Amsterdam the fact that proof of existence has been registered on the blockchain does not proof this registration has been executed by the university.

For this, we need Proof of Origin. Most blockchain implementations support the signing of messages. As an extra security measure the Merkle root can be signed by the University. If we then include the signature within the receipt anyone can verify that this Merlke root has been registered on the blockchain by the University of Amsterdam.

Example implementation on the Bitcoin blockchain

The steps below describe how such a solution could be implemented on the Bitcoin blockchain.

Storing proof of origin & proof of existence on the blockchain:

1) Create a Merkle tree based on any number of input documents, transactions, etc.
2) Create a new transaction with the Merkle root embedded as OP_RETURN message
3) Sign the Merkle root using your private key
4) Create a "receipt" for each document containing:
   a. Each leaf needed to reconstruct the Merkle root
   b. Bitcoin transaction Id
   c. Message signature

Verifying proof of origin & proof of existence

1) Retrieve the sender address & OP_return message from the Bitcoin transaction Id
2) Validate the SHA-256 hash using the sender address & signature
3) Re-construct the Merkle root using the SHA-256 hash from the original document and hashes within the receipt.
4) If the Merkle root can be re-constructed successfully the document is valid.