



Haym Salomon @SalomonCrypto

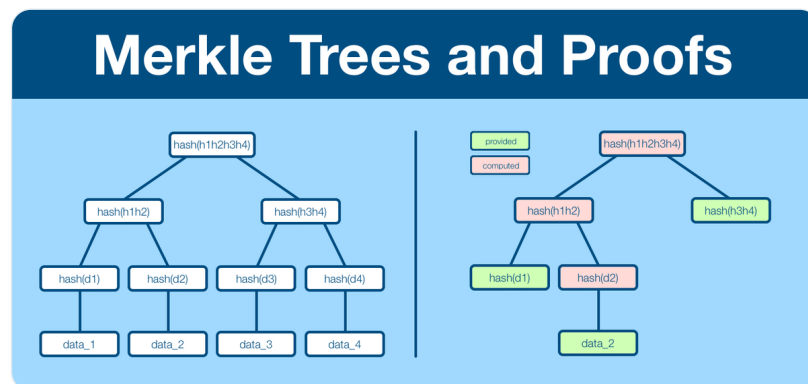
Sep 7 · 14 tweets · [SalomonCrypto/status/1567638108937801728](#)



(1/13) Computer Science 201: Merkle Trees and Merkle Proofs

If you want to understand [@Bitcoin](#), [@ethereum](#) and blockchain technology, you need to learn:

- How a Merkle tree expresses a large dataset
- How a Merkle proof works
- Why a Merkle tree is so efficient



(2/13) Hashing - applying a hash function - takes data (of arbitrary contents and size) and reduces it to a unique, compact string.

Every input produces a unique output, even if two inputs are nearly identical.



The image is a screenshot of a tweet from Haym Salomon (@SalomonCrypto). The tweet is titled "(1/7) Computer Science 101: Hash Functions" and asks "What is a hash function? What are the characteristics of a good hash function? Where do hash functions appear and why do I hear about them all the time?". It includes a link to a guide and a diagram illustrating the concept of a hash function. The diagram shows a table with 'INPUT' and 'OUTPUT' columns, mapping various inputs to their corresponding hash outputs. Below the table, it states: "A good hash function is quick and efficient to compute, but difficult (if not impossible) run in reverse, and distribute values uniformly (randomly) across all possible outputs." The tweet is dated 3:54 PM · Sep 7, 2022 and has 58 likes. It also includes a link to read the full conversation on Twitter and options to reply or copy the link.

Haym Salomon
@SalomonCrypto · Follow

(1/7) Computer Science 101: Hash Functions

What is a hash function? What are the characteristics of a good hash function? Where do hash functions appear and why do I hear about them all the time?

If you want to understand the fundamental tool of crypto, this guide is for you!

Hash Functions

A hash function transforms any amount of data into a compact value of uniform length.

INPUT	OUTPUT
Hello World	0x829bd824b016326a401d083b
Hello Wold	0xabd6bc33983cb06776e89273
Social Security Number: ****-****	0xad22b653d2d85490c0147dfa1
	0x91bfa44d98f1d3e2v2d098d5ff
	0x299c0ce9763c53debb12a87e1

A good hash function is quick and efficient to compute, but difficult (if not impossible) run in reverse, and distribute values uniformly (randomly) across all possible outputs.

3:54 PM · Sep 7, 2022

[Read the full conversation on Twitter](#)

58 Reply Copy link

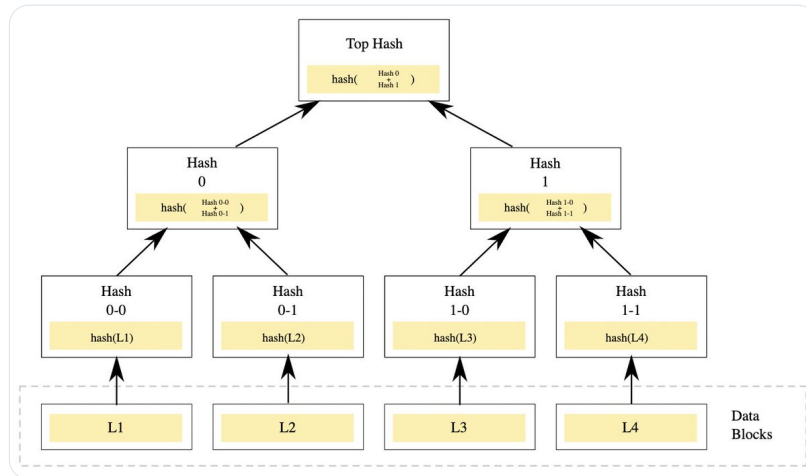
[Read 1 reply](#)

(3/13) A Merkle Tree uses hashing to build a data structure that allows for quick, efficient, verifiable proof that a transaction was included in a much larger data set.

Also called a hash tree, they are named after Ralph Merkle (who proposed them in 1987).

(4/13) A (binary) Merkle tree is a data structure created by first applying a hash function to individual data blocks to create a list of hashes

Then two hashes are combined and hashed again, creating a new (smaller) list of hashes. The cycle repeats until there is a single hash



(5/13) The top hash (also called root hash or root node) is a single value that represents the unique structure of the Merkle tree.

If any of the data is changed, the hash it creates (and all subsequent hashes) will be different, including the root node.

Merkle Trees

A Merkle tree is a data structure that allows the efficient and secure verification of a large set of data without actually transmitting that data.

The tree is constructed first by hashing each piece of data. Then, two (or more) hashes are combined and then hashed again. This continues until all the data has been collapsed into a single hash, also called the top hash or root node.

data₂

→

data₂'

Changing a single piece of data has consequences for the entire Merkle tree. When the new data is passed through the hash function, it produces a different result. Then, this new result is combined with the rest of the hashes, ultimately contributing to the root node.

Each unique dataset has its own unique Merkle tree and root node.

(6/13) The output of a Merkle tree is a unique identifier for that set of data; we can use a hash tree to verify a dataset without transferring the entire dataset

The Merkle root can be derived locally and then compared externally. If the roots match, the data sets are the same

(7/13) Because Merkle trees rely on hashing functions, they only operate in one direction: it is very easy to build a tree from raw data, but it is impossible to recover the data from the tree.

Thus, a Merkle root can be posted publicly without fear of exposing the data.

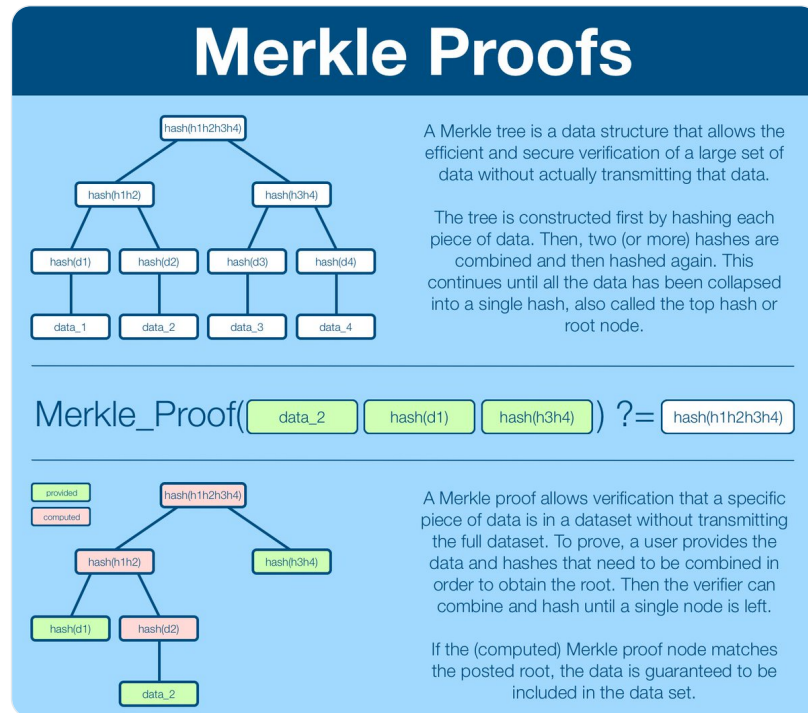
(8/13) However, this has an interesting dilemma: can you confirm if an individual piece of data exists within an Merkle tree?

Put another way, can a Merkle tree do anything other than verify an ENTIRE dataset?

The answer: of course!

(9/13) A Merkle proof is established by providing the specific data and the intermediate hashes which allow a verifier to recreate the Merkle tree.

If the newly computed root node matches root node of the dataset, the verifier can be certain the data is in the dataset.



(10/13) The examples above have been for datasets of size 4, in practice Merkle trees are deployed for datasets in the millions, billions and trillions.

As the dataset get larger, Merkle trees become more and more efficient. Both in coordination and verification.

(11/13) Merkle trees provide a single, unique output for an an entire set of data; a multi-GB file can be reduced down to a single line.

A well designed network can maintain many resources locally and coordinate by communicating an extremely lightweight root hash.

(12/13) This effect could be achieved by simple hashing; a good hash function generates a unique hash for every dataset.


The Merkle proof gives us the ability to verify inclusion incredibly efficiently (for the true nerds, verification complexity scales $O(\log n)$).

(13/13) When you see Merkle trees in the wild, remember this:

Merkle Trees compress huge data sets into a single, encrypted line. Merkle Proofs can be used to efficiently verify data exists in a dataset.

Like what you read? Help me spread the word by retweeting the thread (linked below).

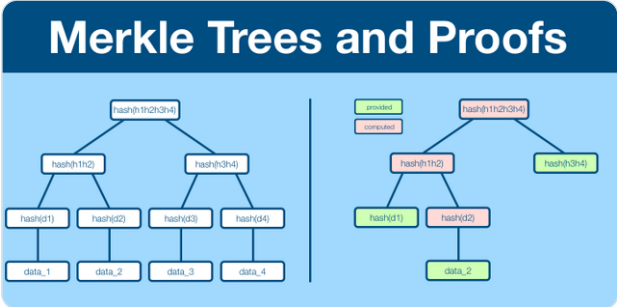
Follow me for more explainers and as much alpha as I can possibly serve.

 **Haym Salomon**
@SalomonCrypto · [Follow](#)

(1/13) Computer Science 201: Merkle Trees and Merkle Proofs

If you want to understand [@Bitcoin](#), [@ethereum](#) and blockchain technology, you need to learn:

- How a Merkle trees expresses a large dataset
- How a Merkle proof works
- Why a Markle tree is so efficient



10:17 PM · Sep 7, 2022

[Read the full conversation on Twitter](#)

193 Reply Copy link

[Read 9 replies](#)

...