Blockchain Council™

Merkle Tree and Hashing

Markle Tree

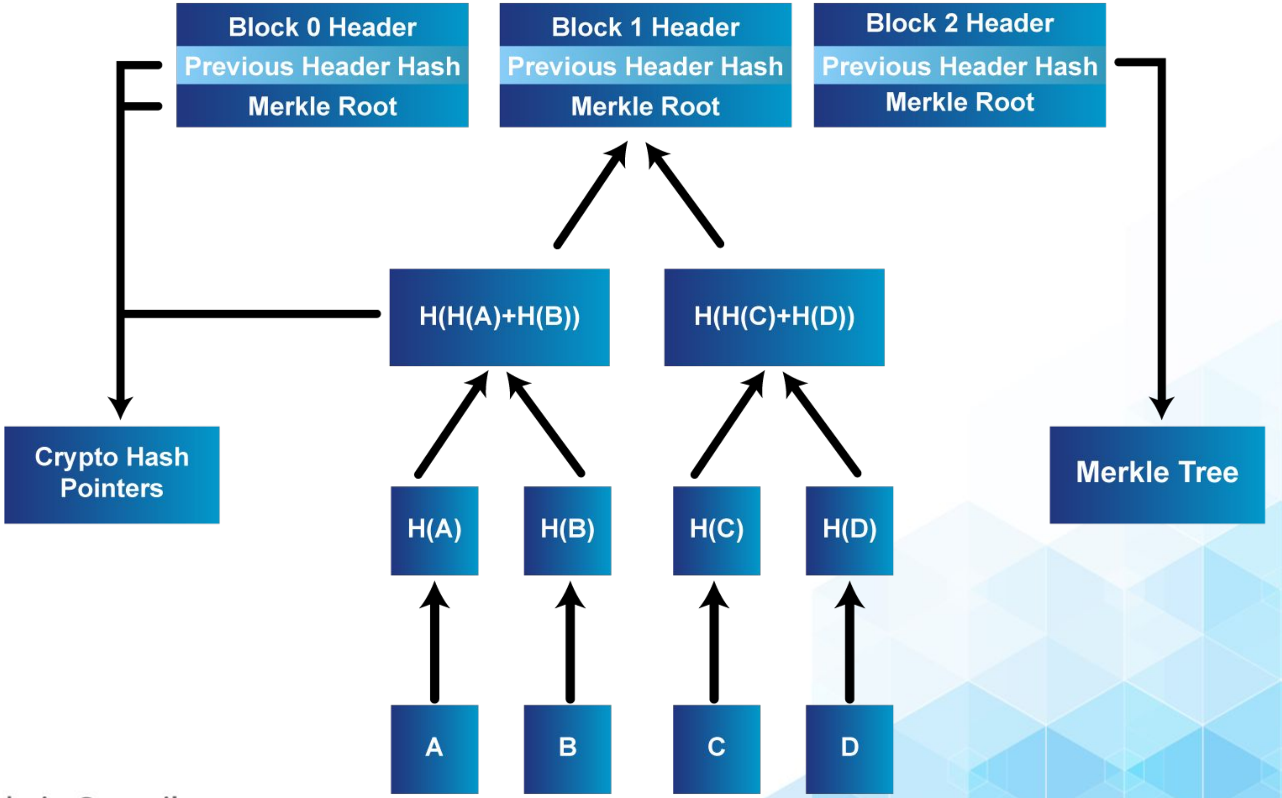
A Merkle tree is a hash-based data structure wherein each leaf node is a hash of a data block, and each non-leaf node is a hash of its offspring. Merkle trees usually have a factor of branching 2, which means that each node has up to 2 children.

The Merkle trees are used for effective data validation in distributed systems. They are secure because instead of using complete files, they use hashes. Hashes are ways to encrypt files that are slightly smaller than the real file.

The verification of integrity is substantially reduced despite of larger data size.

It requires little disk space or memory as the proofs are computationally fast and easy.

Markle Tree


Block 0 Header
Previous Header Hash
Merkle Root
Block 1 Header
Previous Header Hash
Merkle Root
Block 2 Header
Previous Header Hash
Merkle Root
Crypto Hash Pointers
H(H(A)+H(B))
H(H(C)+H(D))
Merkle Tree
H(A)
H(B)
H(C)
H(D)
A
B
C
D

Why Merkle Tree is vital in Blockchain?

Blockchain Council™

For confirming a past transaction, a node would need to reach out to the network in order to get copies of the ledger from its peers.

The node would need to compare each entry line by line.

Any discrepancy between the ledgers, compromise the security of the network.

Every verification request would require large packets of information to be sent over the network.

A lot of processing power is consumed to compare the ledgers, to ensure that there had been no changes.

Hashing

- Hashing is the process of having an input item of any length, converting it into an output item of a fixed length.

- Transactions of different lengths are run through a given hashing algorithm, and all give an output of a fixed length, called as hash.

- Hash size will depend on the hash function used, but the output using a particular hashing algorithm will be of a specific size.

- Cryptographic hash functions are one of the most important techniques in the field of cryptography and are used to accomplish many safety goals such as authentication, digital signatures, generation of pseudo numbers, digital steganography, digital time-stamping, etc.

- Commonly used hashing algorithms is **Bitcoin's Secure Hashing Algorithm 256**, often known as **SHA-256**.