The Magic of Merkle Trees

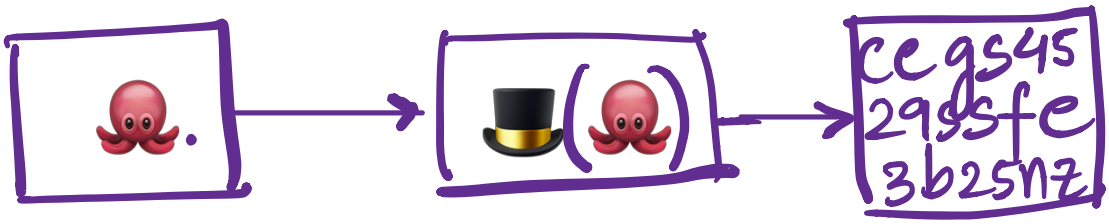
@gauvargi

Introduced in 1979 by Ralph Merkle, Merkle Trees are also known as "Binary Hash Trees".

Merkle trees is a data structure used for efficiently summarizing & verifying integrity of large sets of data.

Before we talk more about Merkle Trees, let's talk about one-way hashing functions.

Hash functions are 1-way functions that take an input & generate a fixed length output.



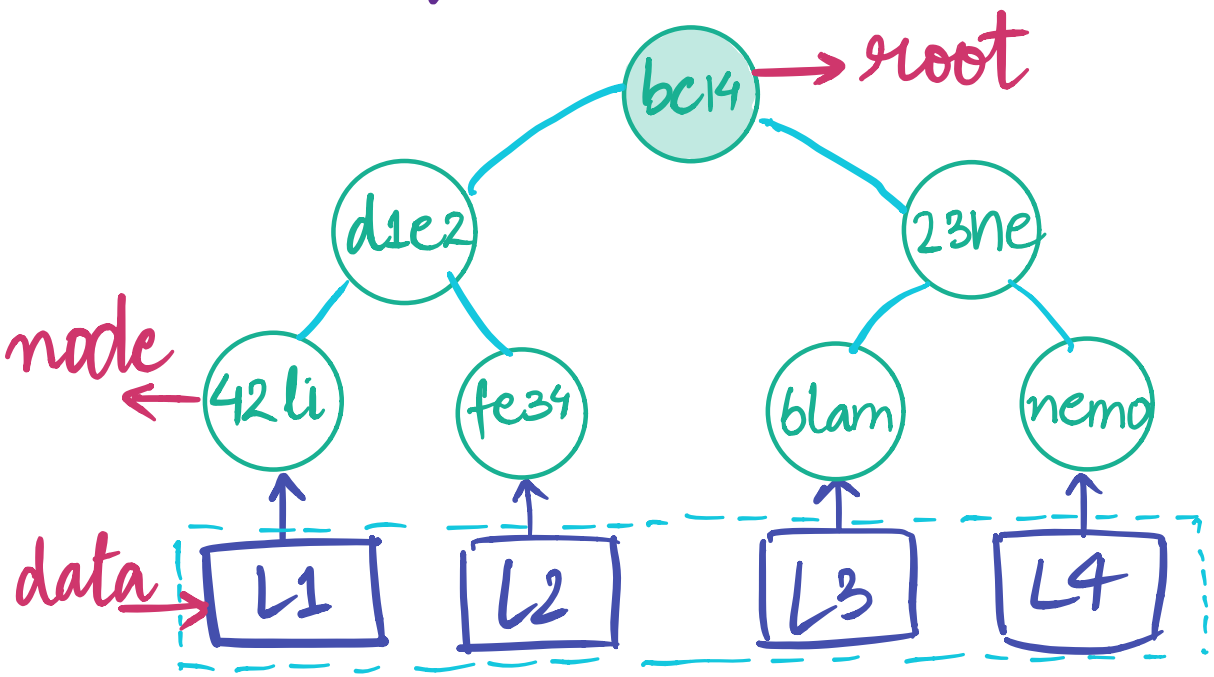
1 way hashing functions are important for:

a) storage: identifying data with fixed length output can create vast storage savings.

b) privacy: if the hash is public, only the calculator of the hash knows the input.

Merkle trees are binary trees where:

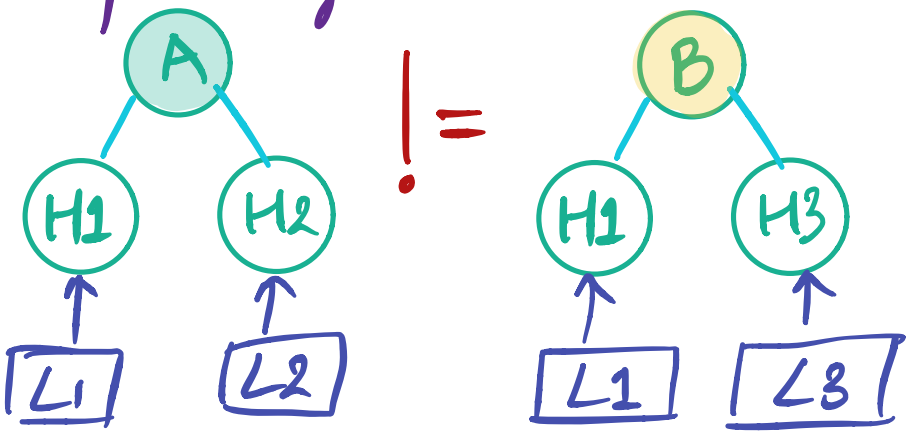
- leaves are hashes of data blocks.
- Nodes contain hashes of their children
- Root hash summarizes the entire data & is publicly distributed.



Usefulness of Merkle Trees

- Detect inconsistencies

If we have two replicas of Merkle Trees, we can compare their equality just by comparing their root hash.

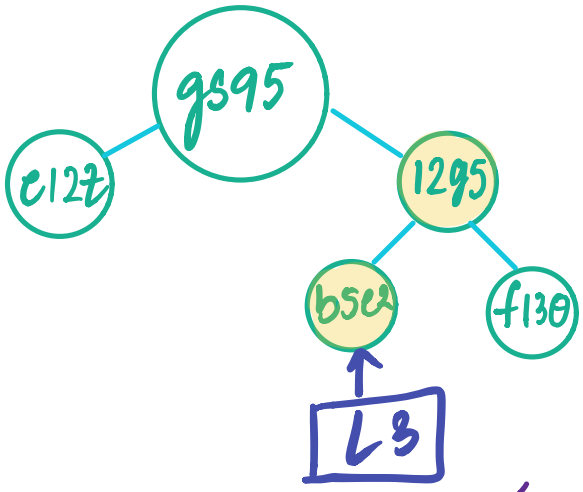


To compare the state of two nodes, the trees are exchanged level by level. If two leaf nodes have different hashes, then objects must be paired.

Used in Dynamo, Cassandra!

- Peer to Peer File Sharing

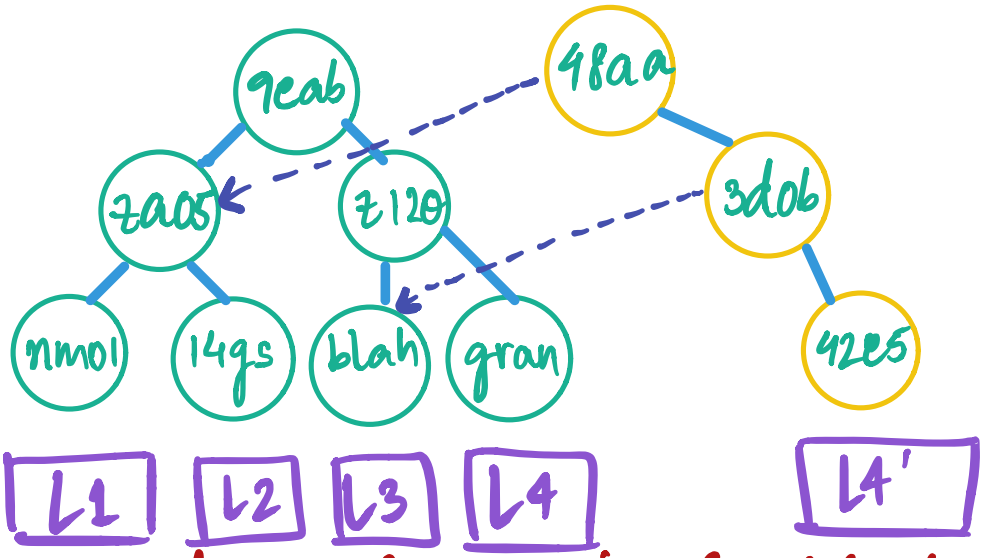
- Fetch the root node from the "trusted" source.



- Fetch L3 & derive its hash, b5e2
- Fetch hash of the right leaf, f130
- We next derive the parent hash, 1295, and gs95 by using the left node hash e127.
- Since the hash is indeed gs95, it confirms that L3 is in the tree.

- Copy on Write

- Copy on write data structures are also called persistent data structures.
- The same tree is shared b/w the copy & the original tree.



Update to a single block L4 (calculate new hashes, rest remain same)

- Only 3 new hashes, all other data blocks are being shared.

Merkle Trees are everywhere!

git, mercurial, IPFS, Cassandra, Ethereum, Bitcoin, etc. all use merkle trees.