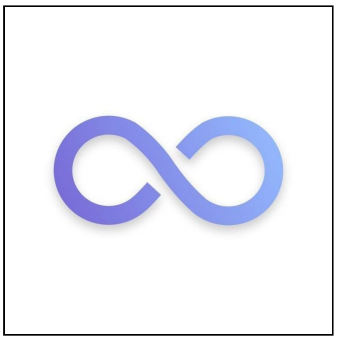
Infnote - Blockchain-based Discussion Forum

A decentralized Information Sharing Platform Based on the Blockchain



First paper has been published.
First release incoming:

- Blockchain Github repository here: https://github.com/Infnote/infnote_chain
- First version of the IOS app also developed

Currently working on:

- Authenticated data structure: Merkle Tree
- Web application

This document is the Merkle Tree work to implement it in Infnote project.
The main purpose of this implementation is to allow light nodes to join the Infnote P2P network. The pages below should briefly demonstrated the features of Merkle tree and why we choose it for Infnote.

Direct connection = Server may send wrong data to light node

- Dictionary data structure: Insert, delete, lookup is Impossible for light node
- Authenticated data structure: Do not need the entire dictionary = Merkel tree

Let's define a merkle tree for blockchain here:
A merkle tree, also known as a binary hash tree, is a data structure used for efficiently summarizing and verifying the integrity of large sets of data. Hashing functions will produce unique values that you can later use to "represent" the inputs. It's important to recognize that these values will exclusively represent the current state of any given input. "State" in this context can represent a number of things. For example, adding a single word to the end of a file or simply changing its contents in any manner. When this happens, the file in question can be said to now hold a new state.

Type of merkle tree:
http://cs.lmu.edu/~ray/notes/orderedtrees/
Bitcoin is using: Binary Merkle Trees because it is only interested in an exchange between sender A and Receiver B.

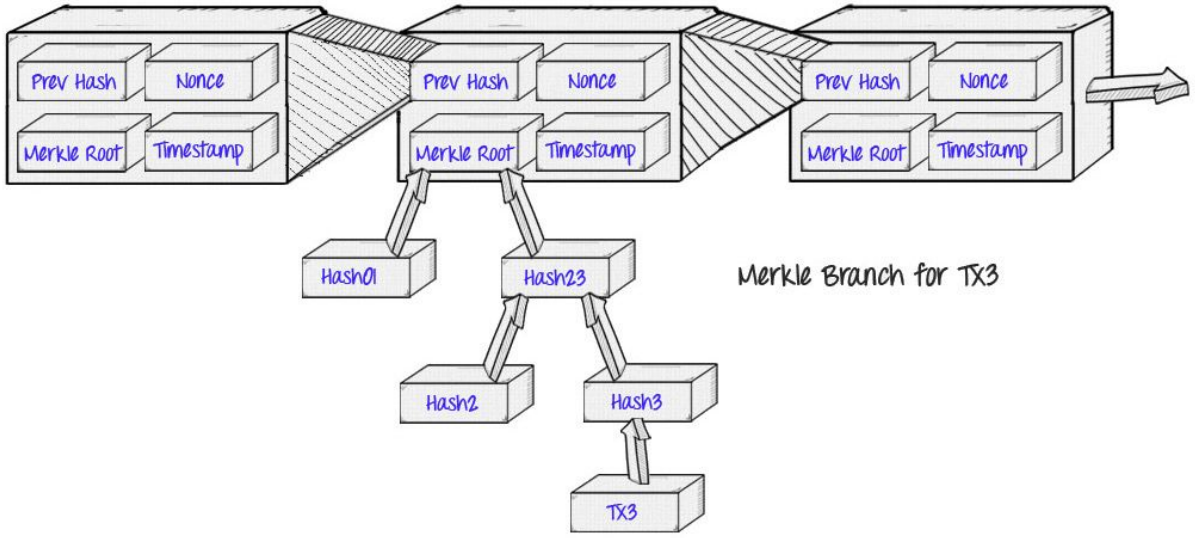
Quick experiment:
echo "Hello from a.txt" >> a.txt && md5 a.txt &&
echo "Hello from b.txt" >> b.txt && md5 b.txt &&
echo "Hello from c.txt" >> c.txt && md5 c.txt &&
echo "Hello from d.txt" >> d.txt && md5 d.txt

*Ouput: c95c68d91441ba192ada81c9cfb2abe7 *a.txt*
*80262782d5961c452eae5de9991af0fd *b.txt*
*0bfe75f719adf6450bb8be8e10126383 *c.txt*
*10fa7f973f6ec6682e1a6f4570a89861 *d.txt*
With a binary tree we would then hash the combination of two siblings (a and b): *md5 <<< "c95c68d91441ba192ada81c9cfb2abe780262782d5961c452eae5de9991af0fd"*
In a binary tree we are combining every siblings (parents on same levels are siblings) until the merkle root.

Merkle Root:
First hash value of the tree, on the top.

Merkle Trees are really important because they allow for "Merkle Proofs". These enable us to quickly verify that a given input has been included in a particular data set — and in what order. This is basically what we are looking for Infnote with the insertion of new articles and posts.
Moreover, merkle trees are especially efficient and allow a lighter data storage.
Indeed, they allow us to compress large data sets by removing all superfluous branches while keeping only the ones we need to establish our proof. Overall performance and scalability is really adapted to blockchain projects, this is again what we are looking for in Infnote project because the light nodes have limited power and storage.

Merkle Proofs in Bitcoin

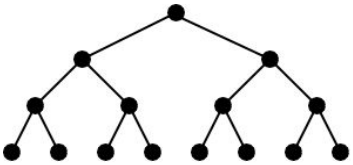


The image above shows that the Bitcoin blockchain uses Merkle proofs in order to store the transactions in every block.
The benefit that this provides is the concept that Satoshi described as "simplified payment verification": instead of downloading every transaction and every block, a light node can only download the chain of block headers, 80-byte chunks of data for each block that contain only five things:

1. A hash of the previous header
2. A timestamp
3. A mining difficulty value
4. A proof of work nonce
5. A root hash for the Merkle tree containing the transactions for that block.

Study of Merkling in Ethereum Blockchain
Patricia Trees

Merkling in Infnote
As a reminder: Infnote projects will involve simple data, indeed the only data is text message for the posts on the different channels.
As a first version, we will consider that the messages will only be sorted by timestamps and by pages.
Binary Merkle trees are very good data structures for authenticating information that is in a "list" format; essentially, a series of chunks one after the other.
As a result, we are defining a first version of our Authenticated data structure as a Binary Merkle Tree.



This figure is a demonstration of the Binary Merkle tree in Infnote. The first node is the Merkle root, then the messages are the last children on the 3rd levels, other levels are the parents used to validate the Merkle proof.

Level	Nodes on Level	Nodes on levels up to and including this one
0	$1 = 2^0$	$1 = 2^{0+1}-1$
1	$2 = 2^1$	$3 = 2^{1+1}-1$
2	$4 = 2^2$	$7 = 2^{2+1}-1$
3	$8 = 2^3$	$15 = 2^{3+1}-1$
4	$16 = 2^4$	$31 = 2^{4+1}-1$
h	2^h	$2^{h+1}-1$

For arbitrary k:

Number of nodes on level h = k^h, obviously

Let S(h) = the number of nodes in a perfect binary tree of height h. Let's find a closed form.

S(h) = 1 + k + k^2 + k^3 + ... + k^h

k × S(h) = k + k^2 + k^3 + k^4 + ... + k^(h+1)

k × S(h) - S(h) = k(^h+1)-1

(k-1) × S(h) = k^(h+1)-1

$$S(h) = \frac{k(h+1) - 1}{k - 1}$$

Source:

https://blog.ethereum.org/2015/11/15/merkling-in-ethereum/
https://medium.com/byzantine-studio/blockchain-fundamentals-what-is-a-merkle-tree-d44c529391d7
http://cs.lmu.edu/~ray/notes/orderedtrees/
https://eprint.iacr.org/2016/994.pdf
https://github.com/sangeeths/merkle-tree
http://www.righto.com/2014/02/bitcoin-mining-hard-way-algorithms.html
https://hackernoon.com/merkle-tree-introduction-4c44250e2da7
https://crypto.stackexchange.com/questions/22669/merkle-hash-tree-updates
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.9700&rep=rep1&type=pdf
http://cs.lmu.edu/~ray/notes/orderedtrees/
https://people.eecs.berkeley.edu/~raluca/cs261-f15/readings/merkle.pdf