ENCRYPTION USING MERKLE TREES

November 6, 2023

NEERAJ KUMAR (2022CSB1095) ,
VARSHAL SAINI (2022CSB1141) ,
SIRI CHANDANA (2022CSB1092)

Instructor:
Dr. Anil Shukla

Teaching Assistant:
SRAVANTHI CHEDE

Summary: A hash tree is a tree of hashes in which the leaves are hashes of data blocks in, for instance, a file or set of files. Nodes farther up in the tree are the hashes of their respective children. For example, in the above picture hash 0 is the result of hashing the concatenation of hash 0-0 and hash 0-1. That is, hash 0 = hash(hash 0-0 + hash 0-1) where "+" denotes concatenation.

---

1. Introduction

A Merkle tree is a hash-based formation utilised in cryptography and computer science.The root hash summarises all data contained in related individual transactions Merkle trees are essential in the reduction of the amounts of data that needs to be maintained in a blockchain for purposes of verification.
A Merkle tree or hash tree, named after the scientist Ralph Merkle, is a hash-based data formation that is used in cryptography and computer science.

What is a merkle tree?

In the Bitcoin network, Merkle trees are used for data verification, which is efficient because hashes are used instead of a complete information file. Demonstrating that a leaf node is a part of a given binary hash tree requires computing a number of hashes proportional to the logarithm of the number of leaf nodes in the tree. Conversely, in a hash list, the number is proportional to the number of leaf nodes itself. A Merkle tree is therefore an efficient example of a cryptographic commitment scheme, in which the root of the tree is seen as a commitment and leaf nodes may be revealed and proven to be part of the original commitment[citation needed].

Working of Merkle trees

A Merkle tree totals all transactions in a block and generates a digital fingerprint of the entire set of operations, allowing the user to verify whether it includes a transaction in the block.
Merkle trees are made by hashing pairs of nodes repeatedly until only one hash remains; this hash is known as the Merkle Root or the Root Hash.
They're built from the bottom, using Transaction IDs, which are hashes of individual transactions.
Each non-leaf node is a hash of its previous hash, and every leaf node is a hash of transactional data.

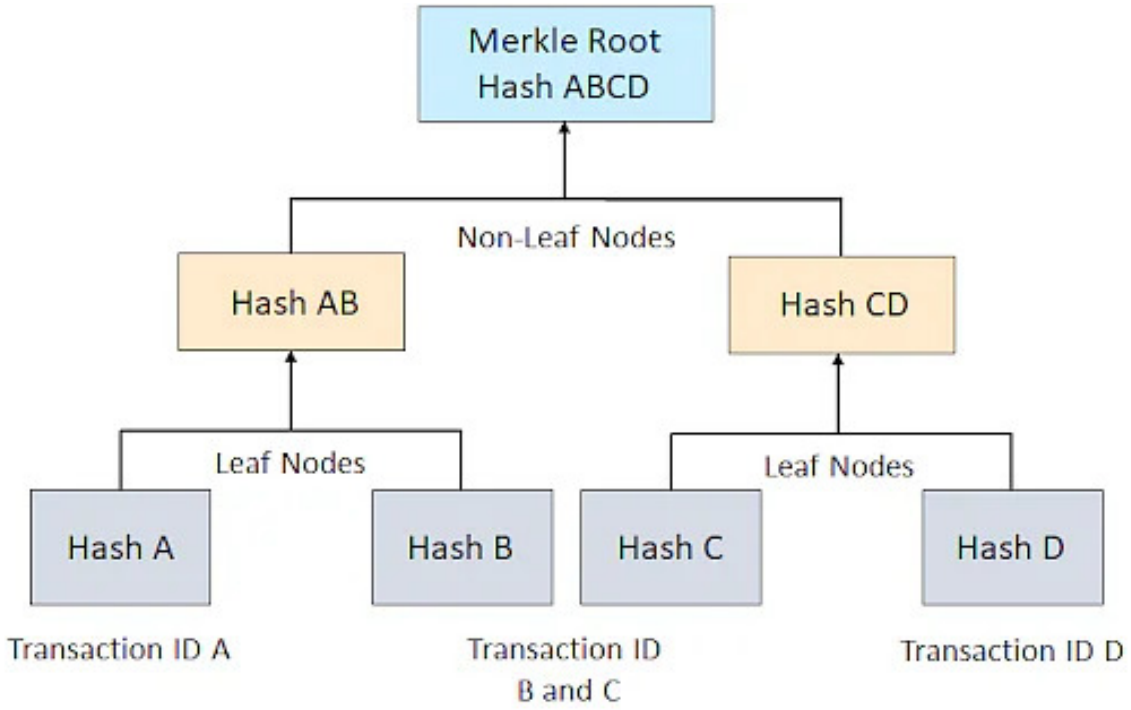

Figure 1: constructing merkle trees

ITS Application in block chain

In bitcoin's blockchain, a block of transactions is run through an algorithm to generate a hash, which is a string of numbers and letters that can be used to verify that a given set of data is the same as the original set of transactions, but not to obtain the original set of transactions. Bitcoin's software does not run the entire block of transaction data—representing 10 minutes' worth of transactions on average—through the hash function at one time, however. Rather, each transaction is hashed, then each pair of transactions is concatenated and hashed together, and so on until there is one hash for the entire block. (If there is an odd number of transactions, one transaction is doubled and its hash is concatenated with itself.)

About MD-5 Algorithm

MD5 is a cryptographic hash function algorithm that takes the message as input of any length and changes it into a fixed-length message of 16 bytes. MD5 algorithm stands for the message-digest algorithm. MD5 was developed as an improvement of MD4, with advanced security purposes. The output of MD5 (Digest size) is always 128 bits. MD5 was developed in 1991 by Ronald Rivest.

Use Of MD5 Algorithm:
1. It is used for file authentication.
2. In a web application, it is used for security purposes. e.g. Secure password of users etc.
3. Using this algorithm, We can store our password in 128 bits format.

Application Of MD5 Algorithm:
1. We use message digest to verify the integrity of files/ authenticates files.
2. MD5 was used for data security and encryption. 3. It is used to Digest the message of any size and also used for Password verification. 4. For Game Boards and Graphics.

Advantages of MD5 Algorithm:
1. MD5 is faster and simple to understand.
2. MD5 algorithm generates a strong password in 16 bytes format. All developers like web developers etc use the MD5 algorithm to secure the password of users. 3. To integrate the MD5 algorithm, relatively low memory is necessary. . It is very easy and faster to generate a digest message of the original message.

Disadvantages of MD5 Algorithm:
1. MD5 generates the same hash function for different inputs.

2. MD5 provides poor security over SHA1.
3. MD5 has been considered an insecure algorithm. So now we are using SHA256 instead of MD5
4. MD5 is neither a symmetric nor asymmetric algorithm.

2. ALGORITHM

Working of the MD5 Algorithm:

1. Append Padding Bits: In the first step, we add padding bits in the original message in such a way that the total length of the message is 64 bits less than the exact multiple of 512.

2. Append Length Bits: In this step, we add the length bit in the output of the first step in such a way that the total number of the bits is the perfect multiple of 512. Simply, here we add the 64-bit as a length bit in the output of the first step. i.e. output of first step = 512 * n – 64 length bits = 64. After adding both we will get 512 * n i.e. the exact multiple of 512.

3. Initialize MD buffer: Here, we use the 4 buffers i.e. A, B, C, and D.
The size of each buffer is 32 bits.
A = 0x67425301
B = 0xEDFCBA45
C = 0x98CBADFE
D = 0x13DCE476

4. Process Each 512-bit Block: This is the most important step of the MD5 algorithm. Here, a total of 64 operations are performed in 4 rounds.
In the 1st round, 16 operations will be performed, 2nd round 16 operations will be performed, 3rd round 16 operations will be performed, and in the 4th round, 16 operations will be performed. We apply a different function on each round i.e. for the 1st round we apply the F function, for the 2nd G function, 3rd for the H function, and 4th for the I function.

We perform OR, AND, XOR, and NOT (basically these are logic gates) for calculating functions. We use 3 buffers for each function i.e. B, C, D.

After applying the function now we perform an operation on each block.
For performing operations we need
1. add modulo (2 raise to the power 32)
2. D[i] – 32 bit message.
3. B[i] – 32 bit constant.
4. «n – Left shift by n bits.

Now take input as initialize MD buffer i.e. A, B, C, D. Output of B will be fed in C, C will be fed into D, and D will be fed into A. After doing this now we perform some operations to find the output for A.

In the first step, Outputs of B, C, and D are taken and then the function F is applied to them. We will add modulo (2 raise to the power 32) bits for the output of this with A.

In a second step, we add the D[i] bit message with the output of the first step. Then add 32 bits constant i.e. B[i] to the output of the second step.

At last, we do left shift operation by n (can be any value of n) and addition modulo by 2 raise to the power 32.

After all steps, the result of A will be fed into B. Now same steps will be used for all functions G, H, and I. After performing all 64 operations we will get our message digest.

3. Figures

Visualized, this structure resembles a tree. In the diagram below, "T" designates a transaction, "H" a hash. Note that the image is highly simplified; an average block contains over 500 transactions, not eight.

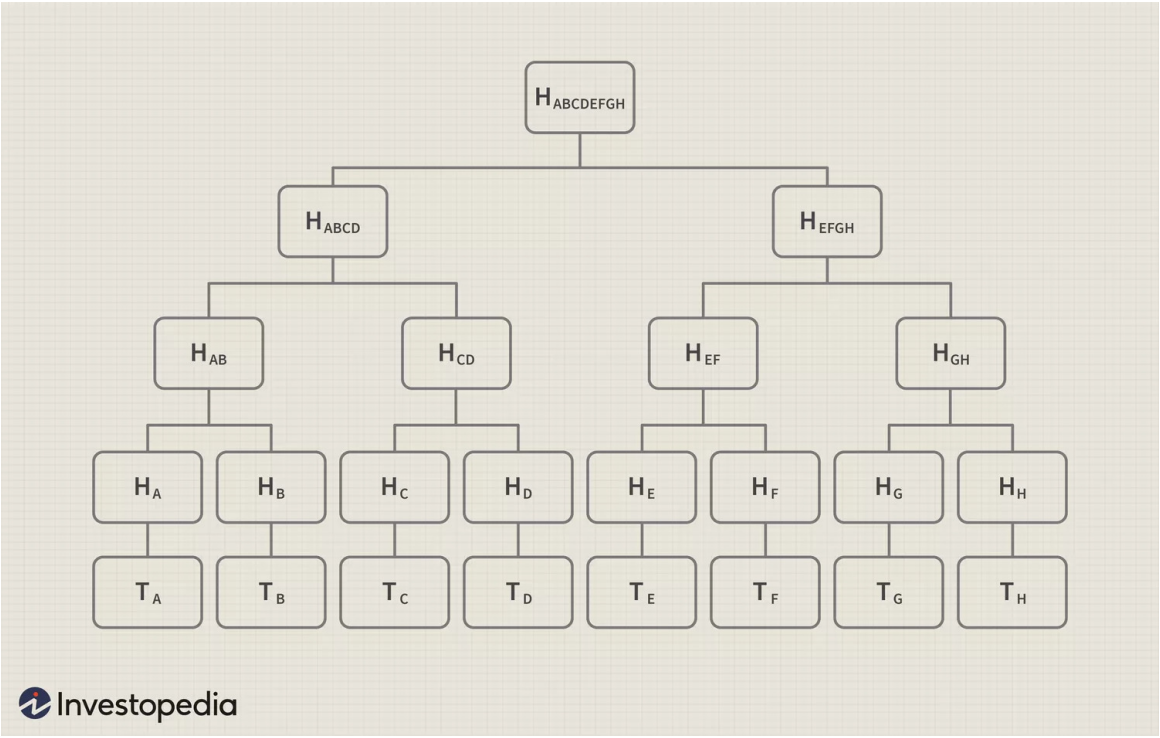

Figure 2: prototype of merkle trees

The hashes on the bottom row are referred to as "leaves," the intermediate hashes as "branches," and the hash at the top as the "root." The Merkle root of a given block is stored in the header: for example, the Merkle root of block 482819 is e045b18e7a3d708d686717b4f44db2099aabcad9bebf968de5f7271b458f71c8.

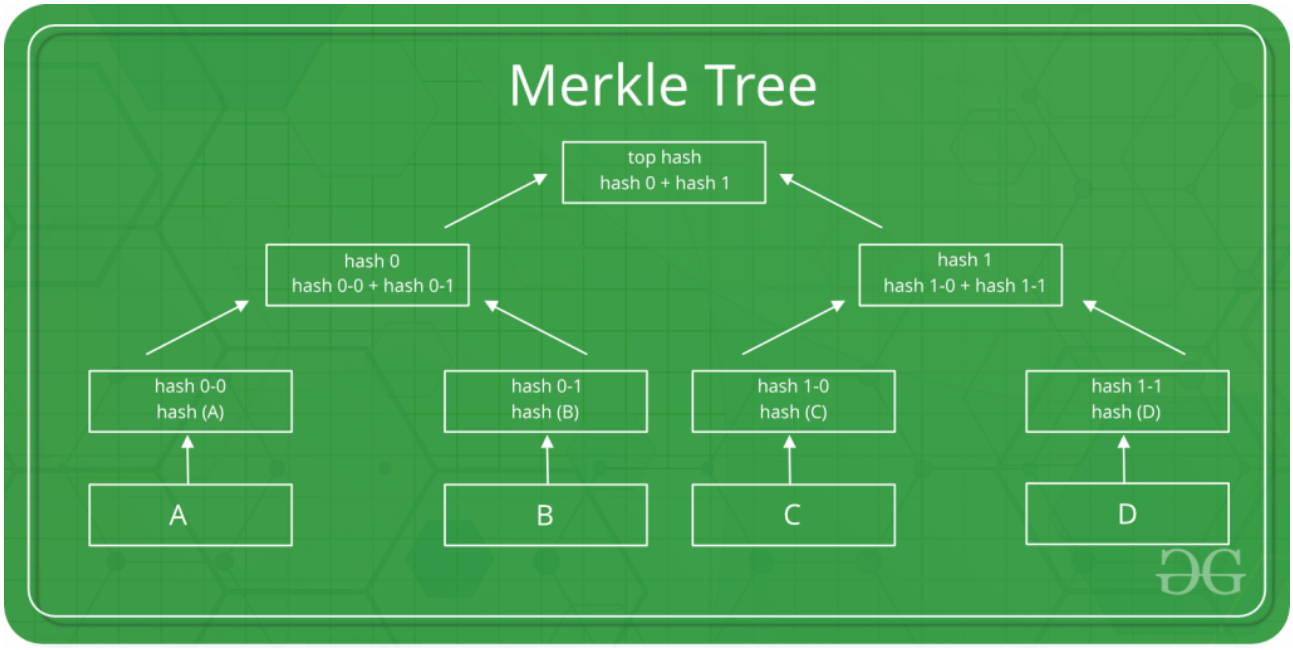

Figure 3: Hashing in merkle trees

4. Conclusions

In recent times merkle trees have gained a lot of popularity due to their wide range of usage in web3.
It plays a crucial role in the security system of blockchain technology and many other peer to peer networks .
It stores data in encrypted form making it nearly impossible for others to retreive the original data from the hash code. Merkle trees in true sense are the backbone of the blockchain security system.

5. Bibliography and References

1. $^{J}ustinChapweske, OnionNetworks, Inc.https://web.archive.org/web/20080316033726/http://www.open-content.net/specs/draft-jchapweske-thex-02.html$

2. $^{Y}i-ChengChen, Yueh-PengChou, Yung-ChenChouhttps://www.researchgate.net/figure/The-architecture-of-Merkle-tree-in-the-blockchain_fig4_334291891$

3. geeks for geeks , https://www.geeksforgeeks.org/blockchain-merkle-trees/

4. Wikipedia https://en.wikipedia.org/wiki/Merkle$_{t}ree$

6. Acknowledgements

I would want to convey my heartfelt gratitude to **Prof. Anil Shukla**, my mentor, for his invaluable advice and assistance in completing my project. He was there to assist me every step of the way, and his motivation is what enabled me to accomplish my task effectively. I would also like to thank our mentor **Ms.Akansha** who assisted me by supplying the equipment that was essential and vital, without which I would not have been able to perform efficiently on this project.
I would also want to thank IIT Ropar for accepting my project in my desired field of expertise. I'd also like to thank my friends and parents for their support and encouragement as I worked on this Project.