2022 Fifth College of Science International Conference of Recent Trends in Information Technology (CSCTIT) | 979-8-3503-3492-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/CSCTIT56299.2022.10145614

An Efficient Bandwidth Based on the Cryptographic Technique of the RSA Accumulator in Block Chain Networks

Kilan M. Hussein
College of Information Engineering
Al-Nahrain University
Baghdad, Iraq
kilan.m.h@uodiyala.edu.iq

M. F. Al-Gailani
College of Information Engineering
Al-Nahrain University
Baghdad, Iraq
m.falih@nahrainuniv.edu.iq

***Abstract*—Blockchain technology is a cryptographic technique that enables users to maintain their data in a decentralized way in a non-editable form without an arbitrator. Data integrity is a significant challenge in blockchain systems. Increasing connectivity will substantially increase the bandwidth and time complexity used by the transaction dissemination or authentication protocol in the blockchain using the Merkle tree, making it prohibitively expensive to operate a full node. The primary goal of this paper is to propose and compare a new transaction authentication technique based on the cryptographic RSA accumulator (ACC) model to the Merkle tree. A proposed ACC model for transaction authentication and proofing has effectively delivered all simulation tests and provided better performance than Merkle tree-SHA256 and Merkle tree-BLAKE256 in blockchain networks.**

Keywords—block chain, cryptographic accumulator, Merkle tree, transaction

I. INTRODUCTION

A block chain is a decentralized network of peer computers that holds a public digital log of transactions (Txs) that is hard to hack or alter. Block chain provides a safe means for persons to communicate with one another without the need for a third party[3-5]. A block chain can also be viewed of as a transactional archive. The storage of transactions is done in blocks. Addresses and information or other details regarding these transactions may be included. A block on the Bitcoin network often holds between 1 to 2500 transactions[6]. These blocks attest to the precise timing and sequence of transactions. The blocks are securely bonded in a way that makes it difficult to alter the data or arrangement of the blocks [7], [8]. Block chain has proven to be particularly effective in a variety of application areas, including, smart contracts, insurance, finance, banking, and a number of other fields, according to researchers[9]-[11].

The blockchain is a publicly accessible, append-only link list data structure that keeps track of all transactions on the network called blocks. Merkle trees are used to record the transactions in each block together with a reasonably secure time date and a hash of the block before it. Fig. 1 displays the process used to create and maintain the blockchain for Bitcoin[12].

Each block in a blockchain contains a few hundred transactions. One must download the whole blockchain in order to verify the transaction's existence in a certain block. Instead, the suggested method can only retrieve a portion of the blockchain's Merkle tree. Transactions obtained through lightweight client are included in the blockchain[1]. This allows lightweight client to verify the hashes while climbing a tree branch. If these hashes match, it is clear that this specific transaction took place in this block ,or in other words,allowing the user to check whether a transaction is included in the block[13]. When a lightweight client examines a specific transaction, such as transaction D in the Merkle tree as showing in Fig. 2,send it to full node and then full node response by send hash C and hash AB. lightweight client can determine whether or not that transaction is included in the block by creating a Merkle path consisting of the hash CD from hashC with hash D, and hash ABCD from hash CD with hash AB and then matching hash ABCD with Merkle root hash.

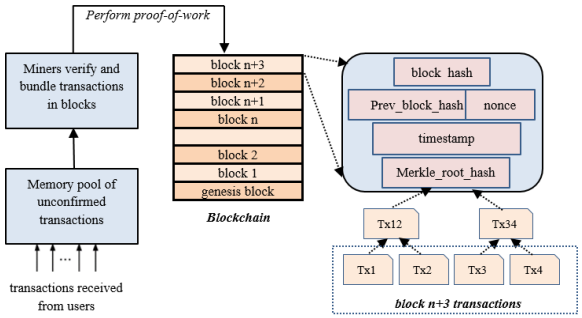

Fig.1. Transactions in block chain[1]

Each transaction will have a hash that must be verified using log2(n), where n is the number of transactions in the block[2].

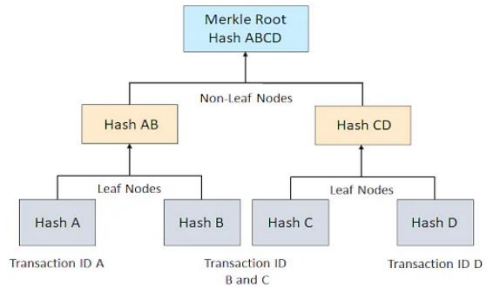

Fig.2. An example of Merkle tree[2]

The bandwidth used by transaction dissemination process will grow significantly as connectivity improves. Bloom filters (BF) offer bandwidth in exchange for a privacy trade-off. But these privacy benefits are only achieved at the cost of some lost bandwidth. Therefore, the false positive rate controls the bandwidth/privacy trade-off which must be carefully tuned[9],[14], [15]. In order to mask the addresses of lightweight nodes, it is possible to specify Bloom filters with a certain false positive probability, which plays an important role in the transactions privacy of users in the Bitcoin network.

The amount of transaction data that needs to be saved in the Merkle tree in contemporary blockchain systems is enormous. When a lightweight node validates the transaction data, all that is required for the full node is to deliver the hash values associated with the data in the specified verification path. This is the current strategy of the Bitcoin network to increase the efficiency of verification. When the information of multiple transactions has to be verified in several blocks, the number of layers in the Merkle tree may differ due to the different number of transactions saved in each block[16].The Merkle proof could result in a significant and costly in time and bandwidth overhead for blockchain applications.

The main contribution of this paper is the suggestion of a new transaction authentication method that makes use of RSA cryptographic techniques.

The remainder of the paper is structured as follows. Section 2 presents related work. In Section 3, the methodology of our research is described. Section 4 performance evaluation of proposed model. Section 5 to evaluate the ACC model in blockchain network. Section 6 presents the conclusion and recommendations for further research.

II. RELATED WORK

The performance of Merkle tree has many prior works and a lot of work is being put into developing various improvements to Merkle trees because user experience in a blockchain system depends on time and space efficiency. U.Chelladurai et al.[17],proposed a new Improved Merkle Tree data structure which can control integrity and guarantee storage security. The efficiency of the suggested method has been tested in the context of healthcare and shown to store and check the authenticity of the content. The binary Merkle Tree data structure has been examined and contrasted with the evaluation metrics of the proposed system. H. Liu et al.[18], provides systematic understanding of Merkle trees in terms of their principles, characteristics, benefits, and applications.S. Jing et al.[19],explore a variety of Merkle tree-based applications, such as those for the Bitcoin and Ethereum blockchain applications, public security audits of cloud computing , image authentication, interplanetary file system and certificate transparency.S. Sato et al.[20],enhancement the implementation of merkle tree by using Plebeia trees.B. Bailey et al.[21], present a thorough examination of Merkle tree-based accumulators, concentrating on factors that minimize the proof size. S. M. Fattahi et al.[22], present the SIMBA blockchain application simulator by adding the Merkle tree functionality to blockchain nodes, SIMBA improves the functionality of an existing simulator and enables more accurate evaluations that aren't achievable with the original tool. Zou et al.[23] , introduce a new strategy called "FAST" to reduce the number of memory access patterns and traversals used to generate a warped Merkle tree for a specific application. Compared to the all-purpose Merkle tree, the proposed method performs better and is more effective.

III. ANALYSIS OF PROPOSED RSA ACCUMULATOR

The definition of the accumulator ACCx is given as ACCx= ACCx=$a^{\Sigma x \in x^x} mod N$, where N is an RSA modulus made up of two very big, secure prime numbers, p and a, chosen at random from the quadratic remaining of N's cyclic group[24].

There are witnesses of the quantity xi given:

$Wit_{xi} \leftarrow acc_x^{xi^{-1}} mod N$, as well as skacc, pkacc = (p, q, N).

Cryptographic technique accumulator can be using as function provided that similar service to a Merkle tree but all transactions (Txs) can be packed together into a one number called accumulator (ACC), given the specific transaction and root in history of block, one then can assemble a witness value which used to verify any transaction in history block in that root. The algorithm as follow:

1. let block contains three transactions
   h- secure hash function of transaction
   N= bits prime number
   a= a generator in (mod N)
   Tx1, Tx2, Tx3 transactions

2. $ACC = a^{h(Tx1)+h(Tx2)+h(Tx3)} mod\ N$ (1)

Add ACC to each header of all blocks in blockchain. After load BF at full node, the full node begins to extracting transactions related to its hash function and send it with its witness (W).

For proof Tx1 witness W=$a^{h(Tx2)+h(Tx3)}$ mod N (2)

3. At lightweight node:
   ACC=$a^{h(Tx1)}$ W mod N (3)
   For proof, ACC equal ACC at lightweight node.

Cryptographic technique accumulator can be using as function provided that similar service to a Merkle tree but all transactions can be packed together into a one number called accumulator (ACC), given the specific

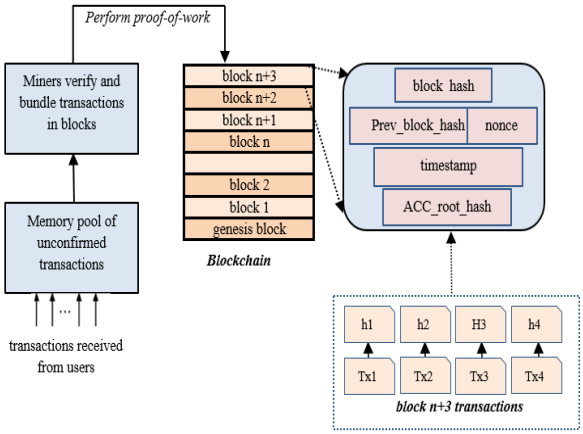

Fig.3. proposed Cryptographic technique accumulator ACC

transaction and root in history of block, one then can assemble a witness value which used to verify any transaction in history block in that root as shown in Fig.3.

Figure 4 shows the simulation steps of the proposed ACC system run at lightweight node and full node. The system models can be executed with various number of elements (E) at lightweight node and various number of universal elements (U) at full node in the blockchain network.

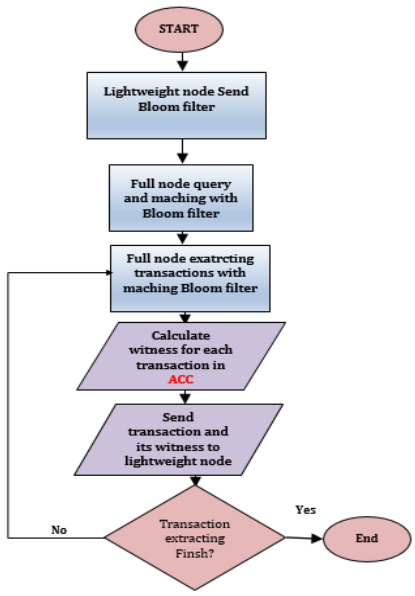

Fig.4. Steps of the proposed ACC model.

IV. PERFORMANCE EVALUATION OF THE ACC MODEL

The simulation was used python 3.8 language win7 and run on an Intel i5 M430 processor running at 2.27 GHz, with 8GB of RAM to implement the Cryptographic technique accumulator as above steps of algorithm and to build Merkle tree. The simulation was implemented several times with (10,20,50) number of Tx that need to proof and (1000) that the average number of Tx in each block.

A. Hash function 256 bits

For ACC we test four big a N=128,256,512 and1024 bits prime number and compute the size of proof information sent using ACC and the size of proof information sent using Merkle tree-SHA256 and Merkle tree-BLAKE256 and compute the computation running time(ms) in each state.

According to "(4)" , the replay of full node needs to count the number of hash functions of a transaction whose content element is authenticated by the Merkle tree. For example, in a full node, if the Bloom filter generates 50 elements with a false positive and the transaction hash size is 32 bytes (hash function size of SHA256=BLAKE2s256) with a block contents 1000 transactions. Then, Blocks × 32 × ($\log_2$ (1000) +1) = 50 × 32 × ($\log_2$ (1000) +1) = 17.6 Kbyte. The bandwidth increases logarithmically according to the number of transactions in each block.

Data size=No. Txs By Bloom filter * size of hash function *(($\log_2$ (Txs in each block) +1) (4)

The results show that ACC need low bandwidth and running time proof than Merkle tree-SHA256, Merkle tree-BLAKE256 and as shown in tables I, II and in Figs. 5 and 6.

TABLE.I. RUNNING TIME (MS) ACC VS MERKLE TREE

Txs	ACC-128	ACC-256	ACC-512	ACC-1024	ACC-2048	Merkle-SHA256	Merkle-BLAKE256
10	10	10	10	30	80	1300	1248
25	30	40	50	80	218	3250	2730
50	60	70	110	156	390	6500	5460

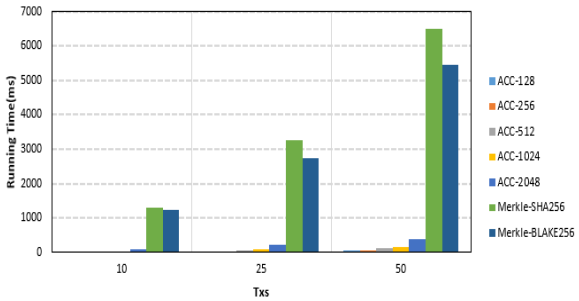

Fig.5. Running time (ms) ACC vs merkle256

TABLE. II.DATA SIZE(BYTE) ACC VS MERKLE TREE

Txs	ACC-128	ACC-256	ACC-512	ACC-1024	ACC-2048	Merkle tree
10	39	77	154	308	617	3509
25	39	77	154	308	617	8772
50	39	77	154	308	617	17545

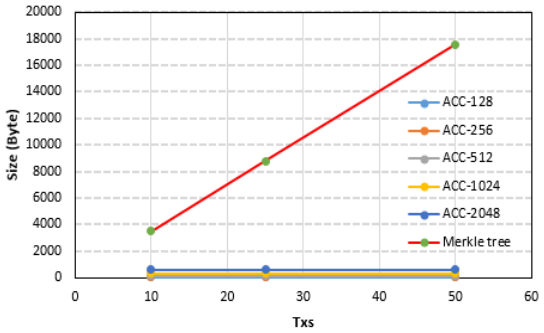

Fig.6. Data size (byte) ACC vs merkle256

B. *Hash function 512 bits*

To figure out how well the proposed ACC model N=128,256,512, and 1024 bits and Merkle tree-SHA512 and Merkle tree-BLAKE512 work, the running time of the computation (in milliseconds) and the size of the data (in bytes) are calculated in each state. "(5)" calculate data size of Merkle tree 512 bits.

Data size =No. Txs By Bloom filter (10,25,50) * size of hash function (byte)*((log$_2$ (Txs in each block) +1) (5)

For example, if Txs=50:

Data size=50* 64 * ((log$_2$ (1000) +1)=35.2Kbyte

The results show that ACC requires less bandwidth and runs faster than Merkle tree-SHA512 and Merkle tree-BLAKE512, as shown in Figs. 7 and 8.

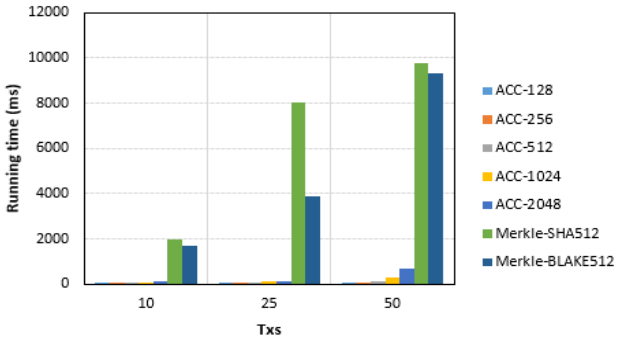

Fig.7. Running time (ms) ACC vs merkle512

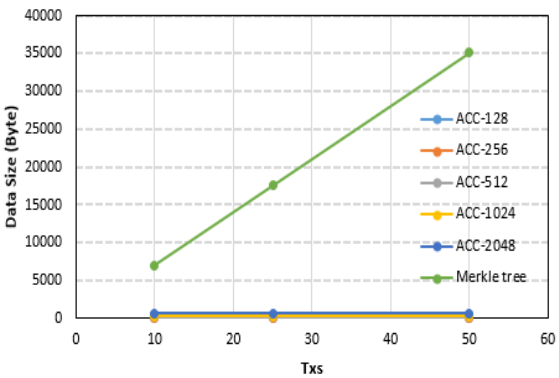

Fig.8. Data size (byte) ACC vs merkle512

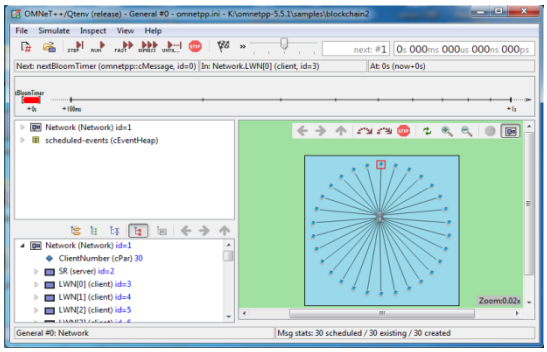

Fig.9. Simulation network

V. SIMULATION ACC MODELWIHT OMNET++

The obtained results based on OMNET++ software simulation have taken place to invistegae and analyze the proposed block chain network that can be applied in an environment that can be controlled[25].This section assesses the proposed authentication transaction ACC model to compare it with the merkle tree model. To evaluate the ACC, a simulation scenario is created by using the server as a full node with 10,000 blocks. Each block contains 1000 transactions(Txs). The proposed scenario is simulated based on different numbers of lightweight nodes from 10 to 100 step 10 as illustrated in Fig. 9. Fig.10 presents the findings surrounding the average response time (RT) of the ACC and compares it with the Merkle tree. ACC provides a response time that is faster than Merkle tree. Fig.10 illustrates that the two algorithms dependent on the number of lightweight nodes in the blockchain network. Fig. 11 compares the throughput (Txs/s) average at a full node for the proposed ACC algorithm and the traditional Merkle tree. where the proposed technique results in greater throughput. Also, the results depend on the ACC parameters and the performance of the full node.

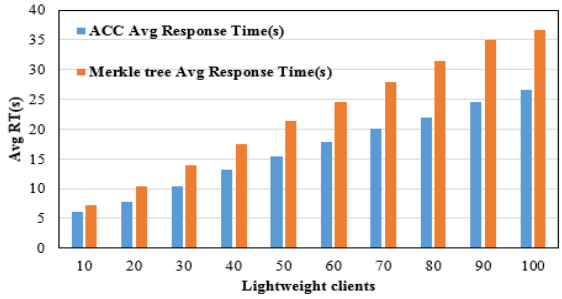

Fig.10. Avg RT (s) ACC vs Merkle tree

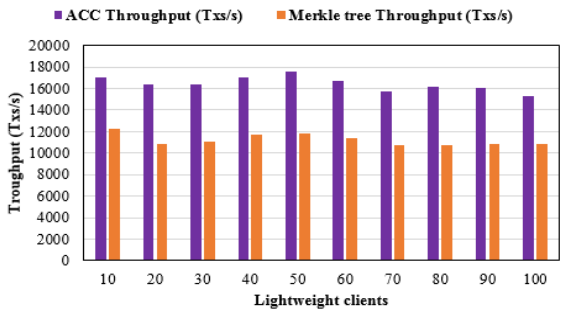

Fig.11. Throughput for ACC and Merkle tree

VI. CONCLUSIONS

A suggested cryptographic method, the RSA accumulator (ACC), for authentication and proofing transactions has passed all simulation tests and outperforms Merkle trees (SHA256) and Merkle trees (BLAKE256) in blockchain networks.

- The processing time of the ACC algorithm was approximately 92 % less than that of the Merkle tree.
- The data size used for authentication and proofing by the ACC algorithm was approximately 93% smaller than that of the Merkle tree.
- The average response times of the ACC algorithm were approximately 27% less than those of the Merkle tree algorithm.
- The ACC algorithm delivered better performance in terms of throughput than the Merkle tree; it was approximately 46% more than that of the Merkle tree with the same bandwidth.

Finally, chaos system can use in future work as an alternative of Merkle tree.

REFERENCES

[1] J. Vijayalakshmi and A. Murugan, "Revamp Perception of Bitcoin Using Cognizant Merkle," 2019. DOI: 10.1007/978-981-13-5953-8_12

[2] S. Dhumwad, M. Sukhadeve, C. Naik, K. Manjunath, and S. Prabhu, "A peer to peer money transfer using SHA256 and Merkle tree," in *2017 23RD Annual International Conference in Advanced Computing and Communications (ADCOM)*, 2017, pp. 40-43: IEEE. DOI: 10.1109/ADCOM.2017.00013

[3] H. Guo and X. Yu, "A survey on blockchain technology and its security," *Blockchain: Research and Applications,* vol. 3, no. 2, p. 100067, 2022/06/01/ 2022.https://doi.org/10.1016/j.bcra.2022.100067

[4] S. Tanwar, "Introduction to Blockchain Technology," in *Blockchain Technology*: Springer, 2022, pp. 1-41. ISBN: 978-981-19-1488-1

[5] A. G. Gad, D. T. Mosa, L. Abualigah, and A. A. Abohany, "Emerging Trends in Blockchain Technology and Applications: A Review and Outlook," *Journal of King Saud University - Computer and Information Sciences,* 2022/03/16/ 2022.https://doi.org/10.1016/j.jksuci.2022.03.007

[6] J. Khamar and H. Patel, "An Extensive Survey on Consensus Mechanisms for Blockchain Technology," 2021, pp. 363-374. DOI: 10.1007/978-981-15-4474-3_40

[7] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*, 2017, pp. 557-564: IEEE. DOI: 10.1109/BigDataCongress.2017.85

[8] Y. Xiao, N. Zhang, W. Lou, Y. T. J. I. C. S. Hou, and Tutorials, "A survey of distributed consensus protocols for blockchain networks," vol. 22, no. 2, pp. 1432-1465, 2020. DOI: 10.1109/COMST.2020.2969706

[9] L. Ge, T. J. S. Jiang, and C. Networks, "A Privacy Protection Method of Lightweight Nodes in Blockchain," vol. 2021, 2021. DOI:10.1155/2021/2067137

[10] D. Dasgupta, J. M. Shrein, K. D. J. J. o. B. Gupta, and F. Technology, "A survey of blockchain from security perspective," vol. 3, no. 1, pp. 1-17, 2019. DOI:10.1007/s42786-018-00002-6

[11] Q. Feng, D. He, S. Zeadally, M. K. Khan, N. J. J. o. N. Kumar, and C. Applications, "A survey on privacy protection in blockchain system," vol. 126, pp. 45-58, 2019. DOI:10.1016/j.jnca.2018.10.020

[12] M. Conti, E. S. Kumar, C. Lal, S. J. I. C. S. Ruj, and Tutorials, "A survey on security and privacy issues of bitcoin," vol. 20, no. 4, pp. 3416-3452, 2018. DOI: 10.1109/COMST.2018.2842460

[13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[14] O. Noam and O. J. A. o. T. Rottenstreich, "Realizing privacy aspects in blockchain networks," vol. 77, no. 1, pp. 3-12, 2022. DOI:10.1007/s12243-021-00861-z

[15] A. Gervais, S. Capkun, G. O. Karame, and D. Gruber, "On the privacy provisions of bloom filters in lightweight bitcoin clients," in *Proceedings of the 30th Annual Computer Security Applications Conference*, 2014, pp. 326-335. DOI:10.1145/2664243.2664267

[16] J. Wang, B. Wei, J. Zhang, X. Yu, and P. K. Sharma, "An optimized transaction verification method for trustworthy blockchain-enabled IIoT," *Ad Hoc Networks,* vol. 119, p. 102526, 2021/08/01/ 2021.https://doi.org/10.1016/j.adhoc.2021.102526

[17] U. Chelladurai, S. J. J. o. A. I. Pandian, and H. Computing, "Hare: A new hash-based authenticated reliable and efficient modified merkle tree data structure to ensure integrity of data in the healthcare systems," pp. 1-15, 2021. DOI:10.1007/s12652-021-03300-y

[18] H. Liu, X. Luo, H. Liu, and X. Xia, "Merkle Tree: A Fundamental Component of Blockchains," in *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*, 2021, pp. 556-561: IEEE. DOI: 10.1109/EIECS53707.2021.9588047

[19] S. Jing, X. Zheng, and Z. Chen, "Review and Investigation of Merkle Tree's Technical Principles and Related Application Fields," in *2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA)*, 2021, pp. 86-90: IEEE. DOI: 10.1109/CAIBDA53561.2021.00026

[20] S. Sato, R. Banno, J. Furuse, K. Suenaga, and A. J. a. p. a. Igarashi, "Verification of a Merkle Patricia Tree Library Using F," 2021. DOI: 10.48550/arXiv.2106.04826

[21] B. Bailey and S. Sankagiri, "Merkle trees optimized for stateless clients in bitcoin," in *International Conference on Financial Cryptography and Data Security*, 2021, pp. 451-466: Springer. DOI: 10.1007/978-3-662-63958-0_35

[22] S. M. Fattahi, A. Makanju, and A. M. Fard, "SIMBA: An efficient simulator for blockchain applications," in *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, 2020, pp. 51-52: IEEE. DOI:10.1109/DSN-S50200.2020.00028

[23] Y. Zou and M. Lin, "FAST: A Frequency-Aware Skewed Merkle Tree for FPGA-Secured Embedded Systems," in *2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2019, pp. 326-331. Doi:10.1109/ISVLSI.2019.00066

[24] Y. Ren, X. Liu, Q. Wu, L. Wang, and W. Zhang, "Cryptographic Accumulator and Its Application: A Survey," *Security and Communication Networks,* vol. 2022, p. 5429195, 2022/03/07 2022. DOI:10.1155/2022/5429195

[25] O. D. E. Simulator. (2022). *Omnet++*. Available: http://omnetpp.org