ESTUDAM Bilişim Dergisi
Cilt 3, Sayı 2, 60-65, 2022

Journal of ESTUDAM Information
Volume 3, Issue 2, 60-65, 2022

DOI: 10.53608/estudambilisim.1086400

(Geliş Tarihi / Received Date: 15.03.2022, Kabul Tarihi/ Accepted Date: 25.05.2022)

(Research Article)

Performance Evaluation of SHA-256 and BLAKE2b in Proof of Work Architecture

Muhammed Mücteba ÖZCAN[1], Burak Alperen AYAZ[2], Muhsin Mert KARAGÖZ[3], Esra Nergis YOLAÇAN*[4]

[1]Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü, 26480, Eskişehir, ORCID No : http://orcid.org/0000-0003-4089-9017
[2]Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü, 26480, Eskişehir, ORCID No : http://orcid.org/0000-0002-5005-6139
[3]Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü, 26480, Eskişehir, ORCID No : http://orcid.org/0000-0001-6135-0985
[4]Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, 26480, Eskişehir, ORCID No : http://orcid.org/0000-0002-0008-1037

Keywords:
Cryptographic Hash,
Blockchain,
BLAKE2b,
SHA-256
Proof of Work

Abstract: The popularity of blockchain, which is a technology with a distributed architecture, is increasing day by day due to the advantages it provides. Along with blockchain, interest in high-performance and efficient hash algorithm applications is increasing rapidly. While considering the amount of power and environmental problems required for these applications, more efficient algorithms were needed. In this study, a comparison of SHA-256 and BLAKE2b, one of the most popular algorithms, is presented. In the experiments, a 4 core Intel i5 with 2.5 GHz frequency based computing system is used. The benchmarking approach focuses on computationally heavy processes such as Proof of Work and Merkle Tree. This article presents a comparison of these two algorithms in a Bitcoin-like mining architecture.

(Araştırma Makalesi)

Emek İspatı Mimarisinde SHA-256 ve BLAKE2b'nin Performans Değerlendirmesi

Anahtar Kelimeler:
Kriptografik Özet,
Blokzinciri,
BLAKE2b,
SHA-256
Emek İspatı

Özet: Dağıtılmış mimariye sahip bir teknoloji olan blokzincirinin, günümüzde, sağladığı avantajlar nedeniyle popülerliği her geçen gün artmaktadır. Blokzinciri ile, yüksek performanslı ve verimli özet algoritma uygulamalarına ilgi de hızla artmaktadır. Bu uygulamalar için gerekli güç miktarı ve çevre sorunları düşünülürken daha verimli algoritmalara ihtiyaç duyulmuştur. Bu çalışmada en popüler algoritmalardan biri olan SHA-256 ile BLAKE2b'nin karşılaştırması sunulmaktadır. Gerçekleştirilen deneylerde, 4 çekirdekli 2.5 GHz frekansa sahip bir Intel i5 tabanlı bilgi işlem sistemi kullanılmaktadır. Kıyaslama yaklaşımı, Emek İspatı ve Merkle Tree gibi hesaplama açısından ağır süreçlere odaklanmaktadır. Bu makale, Bitcoin benzeri bir madencilik mimarisinde bu iki algoritmanın bir karşılaştırmasını sunmaktadır

1. INTRODUCTION

A blockchain is a public digital ledger of transactions held by a network of peer computers in a way that makes it difficult to hack or modify. Blockchain offers a secure way for individuals to compromise directly with each other without an intermediary such as a government or bank.

A blockchain can be described as a historical record of transactions. Each transaction is stored in blocks. These transactions may include information about who, when, where, how much, etc. In the Bitcoin network, a block contains more than 1500 transactions on average [1]. These blocks confirm the exact time and order of transactions. In a way that makes it impossible to change

*İlgili yazar/Corresponding Author: yolacan@ogu.edu.tr

information or order of blocks, the blocks are securely connected. New transactions in the network are collected into the transaction data section of the new block. Each transaction is hashed, paired, and rehashed until a single root hash, called the Merkle root, is left. The Merkle root is stored in the block header. All blocks also store the hash of the previous block's header. Block design in bitcoin is presented in Figure 1.
In the cryptography aspect of blockchain technology, hashing algorithms are used in digital signatures, Merkle trees, consensus algorithms, and blocks. Hash functions are critical to the security of the digital ledger. If a hash function is broken, important hash values (such as the block's chains or a Merkle tree's values) are left completely unprotected. This means that malicious nodes can change the system by running unauthorized software on the blockchain.

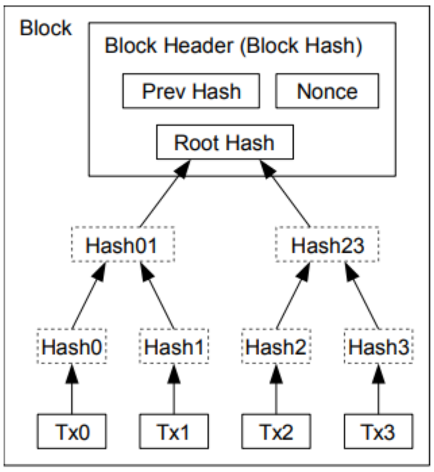


Figure. 1. Block Design in Bitcoin [2]

Since there is no central authority present to validate and verify the transactions in a blockchain, the validation of the ledger is maintained by consensus protocols. The consensus algorithm is a procedure in which all peers of the blockchain network agree about the current state of the network. Consensus mechanisms ensure security in the blockchain network and create trust between unknown peers. There are several commonly used consensus algorithms such as Proof of Work (PoW), Proof of Stake (PoS), Byzantine Fault Tolerance (BFT), and Practical Byzantine Fault Tolerance (PBFT) [3].

PoW is the consensus mechanism included in Bitcoin. PoW network is executed by miners, who are responsible for adding new blocks to the network. In return, these miners receive incentives in the form of block rewards in fixed amounts and transaction fees that paid by users. Miners compete for these rewards by adding computing power to the network; the more computational power, the higher the chance of receiving incentives. This mechanism causes miners to waste huge amounts of energy in the process.

In this work, we develop and demonstrate a PoW based blockchain which uses the BLAKE2b hash function for security. Our purpose is to increase speed and energy efficiency in the software architecture of blockchain. In order to achieve this, we focused on hash functions and observed energy efficiency by comparing 2 different hash functions. With our study, we will make energy use more efficient by directly testing the effect of hash functions on the software architecture of blockchain. For this purpose, we analyze the performance of BLAKE2b compared to SHA-256 hashing algorithm to use in PoW consensus mechanism. Rest of the paper is organized as follows. Related work is presented in Section 2. The methodology of our work is explained in Section 3. Experimental results and discussion are provided in Section 4. Finally, in Section 5, the conclusion and future work is presented.

2. RELATED WORK

In this section, we presented related work on hash algorithms and the effect of different hash functions on the overall performance of the blockchain. Research shows that hash functions appreciably affect blockchain architecture [4]. There are various hashing algorithms on the security market, such as STRIBOG, KECCAK, RIPEMD, BLAKE, SHA (Secure Hashing Algorithm), MD (Message Digest) algorithms [5]. Table 1 shows some algorithms and their usage areas.

Table 1 Usage areas of hashing functions

Hash Function	Usage Areas
BLAKE	Blakecoin cryptocurrency
SHA2	Some government programs
KECCAK	Nexus (NXS), MaxCoin (MAX), Helix Coin (HXC), and so on
STRIBOG	izz.io, BigNet, NWP Solution

In this work, we analyze SHA-256 and BLAKE2 algorithms. The SHA-256 algorithm is a kind of SHA-2, which was published by the National Security Agency in 2001 as a successor to SHA-1 [6]. The SHA-256 hash algorithm takes a message with a length that is smaller than 2^{64} bits as an input and then it produces a 256-bit message digest of the input as an output [7]. SHA-256 is one of the most secure and most used hashing functions among the hash functions [6]. The BLAKE2 family of hashing functions was published in 2012 and designed by Jean-Philippe Aumasson et al [8]. The BLAKE2 family has four hash functions which are BLAKE-224, BLAKE-256, BLAKE-384, and BLAKE-512. Like the SHA-2 algorithm, the BLake has functions for the 32-bit version and 64-bit version as well. These are BLAKE-256 and BLAKE-512, respectively [5]. BLAKE2 design is dependent on the concept of HAIFA structure [8].

There are studies about the development and performance evaluation of blockchain platforms. Xu et al. [9] talked about the software architecture of blockchain and the different factors affecting this architecture. Koteska et al. [10] explored requirements, quality issues, and solutions for a blockchain architecture. These studies show the need for improvement in some aspects of the blockchain platform such as latency, security, cost-effectiveness, privacy, scalability, and performance. A wide field of research focuses on methods to improve blockchain performance. To evaluate blockchain design options, Yasaweerasinghelage et al. [11] presented a simulation and performance modeling framework that predicts the latency of a blockchain-based system. This type of work has proven that changing the parameters of interval and block size is one of the important improvements to solve the latency problem in the blockchain.

Although there is some work involved in evaluating performance of blockchain, to the best of our research, there is not much research on the effect of hash functions on blockchain performance. Among the related works, BLOCKBENCH [12], is the one we find closest to our work. BLOCKBENCH provides benchmarking of various performance metrics across proprietary blockchain platforms such as Ethereum, Hyperledger, and Parity. But BLOCKBENCH has some weak points, such as the inability to generalize for comparison. It is also intended for only a few private platforms. There are a lot of metrics that have been considered about the effects of blockchain performance in BLOCKBENCH. The most critical point for us, the effects of hash functions on these performance metrics have not yet been considered comprehensively. In addition, in the study developed over the BLOCKBENCH framework, the effect of hash functions on the blockchain architecture was examined. Wang et al. [3] chose a private blockchain, Ethereum, for their work. Normally, Ethereum is built on SHA-3, but they also included Ethereum architectures that use various hash functions to expand their research. There are very few studies in the literature on the effects of hash functions on blockchain. When we examine this situation, our main motivation is to measure hash functions and their impacts on the blockchain architecture.

3. MATERIALS AND METHODS

For the comparison of SHA-256 and BLAKE2b in terms of speed, functions should simulate the common processes in new block creation. Since our goal is benchmarking these two algorithms, concepts like memberships, orders, smart contracts are out of this work's scope. We focus on concepts hashing algorithms commonly used in blockchain architectures. To do that, we implement our platform using Python, a high-level programming language, instead of using Ethereum or Hyperledger based networks.

3.1. Hashing Algorithms

We performed our experiments using BLAKE2b and SHA-256 hashing algorithms. BLAKE2b is the successor of BLAKE [13]. BLAKE2b does 12 rounds, and BLAKE does 14 rounds. This difference makes BLAKE2b about %25 faster than its ancestor [14]. To compute BLAKE2b in our work, we use the Python hashlib module.

SHA-256 does 64 rounds, quite a lot compared to BLAKE2b. SHA-256 uses variable-length messages that are divided into 512-bit blocks and 256-bit key length as BLAKE2b. To compare to BLAKE2b, SHA-256 implementation in the Python hashlib module is used.

3.2 Consensus Mechanism

Consensus algorithms can be defined as mechanisms that enable a decentralized network to make unanimous decisions when it is necessary [15]. In this work, PoW is selected as the consensus mechanism. PoW is a cryptographic proof that the miner node proves that a certain amount of computational power has been spent [16]. This is one of the most popular consensus algorithms where hashing algorithms are heavily used. Because of its common usage, we use PoW to compare BLAKE2 and SHA-256, similar to one in the Bitcoin network. The puzzle of our PoW is "*find a number x that when hashed with the previous block's solution,* a *hash with n leading 0s is produced* ". The miner node will try to find a *proof x* that hashing with previous proof will produce a 256-bit length hash with *n* leading zeroes. Several leading zeros *n* is a variable that differs in each round of comparison.

3.3 Data Structure

Merkle tree is a type of data structure that holds the root hash of all transactions in a block. A Merkle Tree (presented in Figure 2) totals all transactions in a block and generates a hash of the entire transaction, allowing the user to verify whether it includes a transaction in the block. It is used to validate data integrity efficiently. To compare SHA-256 and BLAKE2b effectively, transactions per block is a variable that changes in every round.

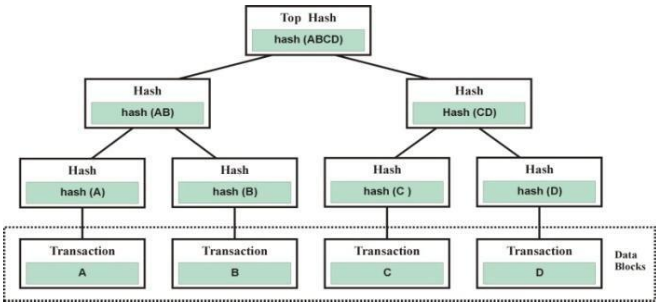


Figure 2. An Example of Merkle Tree [17]

3.4 Blockchain Architecture

A blockchain that includes functions to create a new transaction and a new block need to be implemented. The blockchain architecture to be used includes PoW and Merkle Tree to compare the performance of BLAKE2b and SHA-256. For each hashing algorithm, various implementations of the proposed architecture in Python are used as presented in Figure 3.

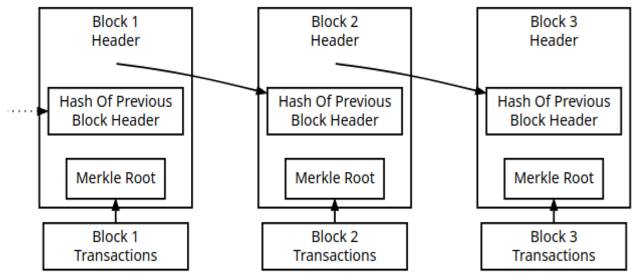


Figure 3. A high-level diagram of the proposed architecture.

In order to make transactions and interact with the proposed chain, an HTTP API (Application programming interface) needs to be implemented. There are 3 endpoints which are for new transactions (POST), mining operations (GET), and monitoring the chain (GET). For HTTP operations, FastAPI, a Python web framework, is used. As a client requests, a Python HTTP library is used.

3.5 Test Environment Specifications

Since BLAKE2b is optimized for 64-bit architectures [14], we use a 64-bit CPU to get the best performance. The technical specifications of the machine used for testing are following:

- Intel i5-7300HQ, 4 cores, 2,50 GHz, CPU
- 2x DDR4, 8192 MB, 2400 MHZ RAM
- NVIDIA GeForce GTX 1050 Graphics Card
- 256 GB SSD
- Arch Linux with Kernel 5.11

To perform our experiment, two hashing algorithms are compared in different scenarios. In each round, the maximum length of the blockchain is 10 blocks, and the running time for a block of two algorithms is measured. There will be 3 different *puzzles* for PoW: 2, 4, 6 leading zeros, and 2, 4, 6 transactions per block for each different puzzle. In total 9 different configurations is used. Official implementation of our work can be accessed on https://github.com/mucozcan/sha-blake-benchmark.

4. RESULTS

There are 9 different cases defined to test performances of SHA-256 and BLAKE2b. Each case is slightly more complex than the previous case. In total 18 cases were executed (9 for each algorithm). All the transactions made in block operations were identical. Configurations can be seen in Table 2.

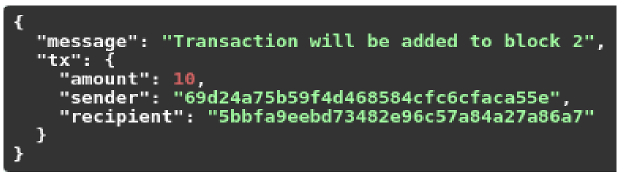
Table 2 Test Cases

Round	Puzzle	Transactions	Number of
#1	2	2	10
#2	2	4	10
#3	2	6	10
#4	4	2	10
#5	4	4	10
#6	4	6	10
#7	6	2	10
#8	6	4	10
#9	6	6	10

An automated client is implemented for the testing process. Using Python's requests module, a script has been written to make requests on certain defined endpoints. There are 3 endpoints in total to create transactions, trigger mining operations, and view the chain's current state (presented in Table 3).

Table 3 API Endpoints

HTTP Method	URI Path	Description
GET	*/chain*	Retrieves the current state of the chain
GET	/mine	Triggers mining operation with current transactions
POST	*/tx/new*	Adds a new transaction



```
{
  "message": "Transaction will be added to block 2",
  "tx": {
    "amount": 10,
    "sender": "69d24a75b59f4d468584cfc6cfaca55e",
    "recipient": "5bbfa9eebd73482e96c57a84a27a86a7"
  }
}
```

Figure 4. Transaction response

An example of a returned response after creating a transaction can be seen in Figure 4. After sending required transaction requests for a block, a mining request is sent to the server. Measuring time for a given hashing algorithm to mine is started immediately after a request has been received and finished after a block is created. Firstly, PoW runs and returns a proof and a nonce. Creation of a transaction of miner reward is created immediately after. A Merkle root is created with all pending transactions. Using this Merkle root, previous hash, and nonce, a new block is created and the proof is started. After creating a block successfully, the elapsed time from start is taken in nanoseconds, and timing results are saved to a file and a response is returned. For a given single plaintext, BLAKE2b is about 2 times faster than

SHA-256, in the proposed test environment. The result is presented in Figure 5.

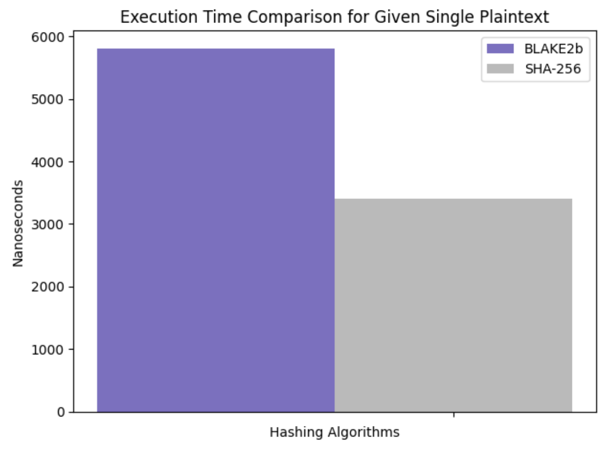


Figure 5. Execution time comparison for a given single plaintext.

After 18 rounds, 90 mining operations for each hashing algorithm, results are compared. In the first 3 rounds, there was very little difference between SHA-256 and BLAKE2b. In Figure 5, elapsed time is given in nanoseconds for visual purposes. Since it was a single iteration process, it took much less time than other testing processes.

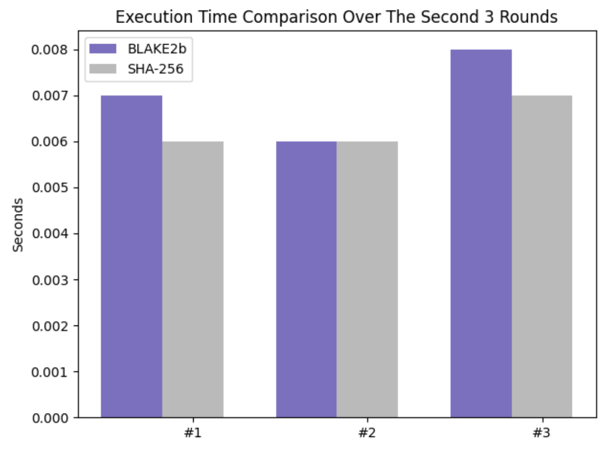


Figure 6. Execution time comparison over the first 3 rounds.

There are also some inconsistencies in measured times due to the *randomness* in the number of iterations required for mining a block. Some mining operations can be done more quickly or slowly depending on the input hashes, even with the same transactions. Figure 7 shows that in round 6 BLAKE2b based PoW runs too many iterations compared to SHA-256 one.

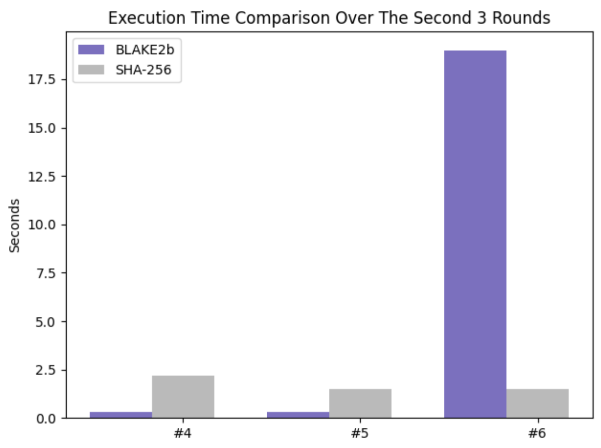


Figure 7. Execution time comparison over the first 3 rounds.

Since PoW contains a puzzle, finding *proof* to get desired hash with given inputs, it's an algorithm with a high computational cost. BLAKE2b has a 64-byte digest size while SHA-256 has a 32-byte. Because of this difference, getting valid nonce using BLAKE2b is about 2 times slower compared to SHA-256, assuming the same number of iterations are done in PoW. Figure 8 shows that SHA-256 outperforms BLAKE2b in harder puzzles (must be 6 leading zeros in nonce) rounds.

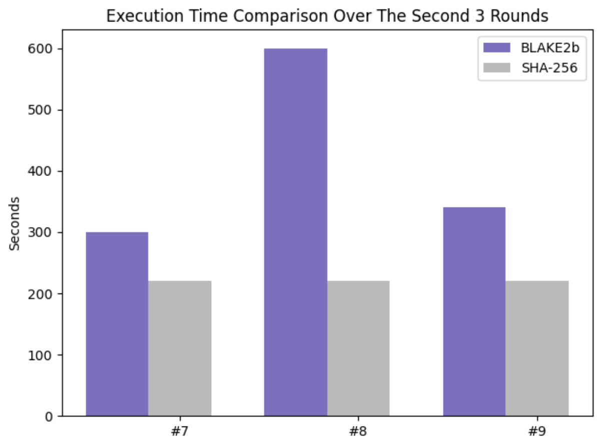


Figure 8. Execution time comparison over the first 3 rounds.

As can be seen in results of 9 test cases, our study shows that BLAKE2b is faster than SHA-256 for hashing a single plaintext but for complex and computationally expensive algorithms like PoW, SHA-256 outperforms BLAKE2b due to the smaller digest size.

4. DISCUSSION AND CONCLUSION

Today, blockchain has started to be used in almost every field, especially where data needs to be tracked or accessed. Considering the nature of the blockchain, the copy of the data must be located and processed on all nodes, the first of its advantages is that the data can be processed within the consensus. In addition to the advantages provided by this structure, the computing speeds of blockchain networks with various architectures and mechanisms have also gained importance. Hashing algorithms have been considered to have an important place in this sense. This article presents a comparison of these two algorithms in a Bitcoin-like mining architecture. Even though BLAKE2b is faster than SHA-256 for hashing a single plaintext, SHA-256 performs much faster operations on a complex architecture like PoW. To be able to take advantage of the efficiency of BLAKE2b, another consensus algorithm than Proof of Work should be tried for further studies.

REFERENCES

[1] Drijvers, M., Gorbunov, S., Neven, G. and Wee, H., 2020, August, Pixel: Multi-signatures for Consensus, In USENIX Security Symposium, 2093-2110.

[2] Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review, p.21260.

[3] Wang, W., Hoang, D.T., Hu, P., Xiong, Z., Niyato, D., Wang, P., Wen, Y. and Kim, D.I., 2019. A survey on consensus mechanisms and mining strategy management in blockchain networks. Ieee Access, 7, 22328-22370.

[4] Wang, F., Chen, Y., Wang, R., Francis, A.O., Emmanuel, B., Zheng, W. and Chen, J., 2019. An experimental investigation into the hash functions used in blockchains. IEEE Transactions on Engineering Management, 67(4), pp.1404-1424.

[5] Kuznetsov, A., Shekhanin, K., Kolhatin, A., Kovalchuk, D., Babenko, V. and Perevozova, I., 2019, December. Performance of hash algorithms on GPUs for use in blockchain. In 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT) (pp. 166-170). IEEE.

[6] SHA-256 Algorithm Overview, available online: https://www.n-able.com/blog/sha-256-encryption

[7] Rachmawati, D., Tarigan, J.T. and Ginting, A.B.C., 2018, March. A comparative study of Message Digest 5 (MD5) and SHA256 algorithm. In Journal of Physics: Conference Series, Vol. 978, No. 1, p. 012116, IOP Publishing.

[8] Atiwa, S., Dawji, Y., Refaey, A. and Magierowski, S., 2020. Accelerated hardware implementation of blake2 cryptographic hash for blockchain. In 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 1-6.

[9] Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., Pautasso, C. and Rimba, P., 2017, April. A taxonomy of blockchain-based systems for architecture design. In 2017 IEEE international conference on software architecture (ICSA), 243-252, IEEE.

[10] Koteska, B., Karafiloski, E. and Mishev, A., 2017, September. Blockchain implementation quality challenges: A literature. In SQAMIA 2017: 6th workshop of software quality, analysis, monitoring, improvement, and applications, Vol. 11, 2017.

[11] Yasaweerasinghelage, R., Staples, M. and Weber, I., 2017, April. Predicting latency of blockchain-based systems using architectural modelling and simulation. In 2017 IEEE International Conference on Software Architecture (ICSA), 253-256.

[12] Dinh, T.T.A., Wang, J., Chen, G., Liu, R., Ooi, B.C., and Tan, K.L., 2017, May. Blockbench: A framework for analyzing private blockchains. In Proceedings of the 2017 ACM international conference on management of data, 1085-1100.

[13] Aumasson, J.P., Henzen, L., Meier, W. and Phan, R.C.W., 2010. Sha-3 proposal blake. Submission to NIST (2010). URL http://131002. net/blake/blake. pdf, 495.

[14] Aumasson, J.P., Neves, S., Wilcox-O'Hearn, Z. and Winnerlein, C., 2013, June. BLAKE2: simpler, smaller, fast as MD5. In International Conference on Applied Cryptography and Network Security, 119-135.

[15] Sankar, L. S., Sindhu, M., and Sethumadhavan, M., 2017. Survey of consensus protocols on blockchain applications. In 2017 4th international conference on advanced computing and communication systems (ICACCS), pp.1-5..

[16] Meneghetti, A., Sala, M. and Taufer, D., 2020. A survey on pow-based consensus. Annals of Emerging Technologies in Computing (AETiC), Print ISSN, pp.2516-0281.

[17] Bosamia, M. and Patel, D., 2018. Current trends and future implementation possibilities of the Merkel tree. International Journal of Computer Sciences and Engineering, 6(8), pp.294-301.