4th International Conference on Current Research in Engineering and Science Applications (ICCRESA 2022), Alsalam University College, Baghdad, Iraq

2022 4th International Conference on Current Research in Engineering and Science Applications (ICCRESA) | 979-8-3503-3494-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCRESA57091.2022.10352512

Cryptocurrencies in Blockchains Environment: The Verification Trip

Marah Mohammed Taha
Department of Computer Science,
AL Noor Unversity College
Mosul, Iraq
Marah.Mohammed@alnoor.edu.iq

Mafaz Alanezi
Department of Computer Science,
University of Mosul
Mosul, Iraq
mafazmhalanezi@uomosul.edu.iq

***Abstract*—A peer-to-peer network called a blockchain, a secure ledger of transactions, such as buying, selling, and transferring, is in charge of organizing and managing the Cryptocurrencies. In contrast with physical money, they are decentralized, which implies neither governments nor other financial organizations issue them. In a cryptocurrency blockchain, transaction data is permanently preserved. A block contains some or all of the most recent transactions the network still needs to confirm. The block is closed once the data has been confirmed. A series of verifications are applied to the transactions that blockchain participants submit to ensure their integrity and for most network security standards to be met. In this paper, we will discuss a set of them.**

Keywords*— *Blockchain, Cryptocurrencies, Verification, Hash, Algorithm

I. Introduction

The source code, technology, and idea of what is now known as the blockchain were explained in a white paper written in 2009 by an unidentified person named Satoshi Nakamoto, where he introduced the original cryptocurrency known as Bitcoin [1]. With billions of dollars spent on it, Bitcoin has skyrocketed in popularity and is presently the most successful digital currency worldwide. It has been demonstrated to be highly secure in terms of networks and protocols [2]. Blockchain technology allows data into such a ledger without needing a central organization. Several independent persons with technical competence cobble together a ledger connecting data items with their owners, and before accepting and transmitting the included transactions, each node verifies them all. Eventually, the system state that all nodes concur through the consensus mechanism is saved on all [3]. The decentralized consensus arises from the interaction of many activities that take place independently across the network's nodes: independent verification of each transaction, independent accumulation of those verified transactions into new blocks, corroboration of each new block by every node and assembly it into the chain, and so on [1]. The 1st part of this paper includes an explanation of the blockchain, which is the infrastructure of cryptocurrencies, with a group of related works. In comparison, the main idea of this research is centered in the 2nd and 3rd parts, which are the formulation and discussion of a set of verification stages that transactions are exposed to until they are accepted and considered part of the network data.

II. Primitive Knowledge and Related Works:

Blockchain is a type of distributed ledger that records transactions. Blocks are created and sorted according to the transactions, where each transaction denotes a value transfer from one address to another [4]. Blockchains integrate various levels of digital encryption technology and user identities. Additionally, algorithms are used to resolve complex mathematical puzzles and reach agreements on the veracity of the user network [5]. The block is a collection of transactions identified by the fingerprint and timestamp. The transactions are verified via producing proof-of-work by hashing the block header, where proof-of-work is the most well-known consensus technique implemented for the first time in bitcoin. It requires that miners discover a numerical solution to the SHA256 algorithm that satisfies the target and difficulty of the network [1]. Vallois and Guenane offered an analysis of bitcoin's validity: transaction version, signature validity, header validity, tree validity, etc. They also described how the procedure is linked to consensus and how it is intended to facilitate node synchronization. They observed that verification requires more processing power than searching a database, which is the most expensive part of the validation process.

Nevertheless, bitcoin is still slow enough for each node to have time to complete sufficient verifications before getting a new block [6]. Shih et al. proposed smart contracts for mining verification built into the Ethereum platform to solve the problem of the "cryptojacking" attack. They built "SiteManagement" and "VerifiedSite" smart contracts to address the issue of web mining certification and examined several contract security concerns [7]. Taher et al. provided an extra layer in user verification to develop the cryptocurrency's security by using a "multi-factor authentication" algorithm with the TOTP technique, so the username and password are required as a first factor, followed by the MFA token, which serves as a second factor to create a TOTP. It has been discovered that MFA can offer a more effective method for securing cryptocurrency transactions [8]. Lee et al. provided an algorithm to prevent "double spending attacks," Each transaction must start as "Unapproved," keeping senders waiting for receivers. Only the recipient's private and unique public keys can be used to establish the transaction's state and propagate it. The receiver is proven if the produced public key perfectly matches the public key used in the transaction, in other words. Using this approach, the sender cannot spread out several double-payment transactions over the network because the receiver can control the propagation time following the approval due to a sequence that prevents duplicate spending [9].

III. VERIFICATION PHASES

Each block in the chain comprises the data, the preceding hash, and the hash to that specific data. Depending on the type of blockchain, many data are kept there. If it is linked to bitcoin, the blockchain will contain transaction data, sender and recipient information, and the total amount of bitcoin in circulation. The hash value is a crucial consideration when making changes to the block. A block will not be regarded as belonging to the same block if its hash value changes. The block also contains the hash of the one before it in addition to the current block's hash. Connecting the present block to the preceding block aids in forming a chain. Blockchain is more secure due to these characteristics of a block in the chain [10]. In this section, we'll show a collection of Python-built data verification procedure algorithms where the blockchain setup is a cryptocurrency created specifically for this study's testing purposes.

__init__(): Function initials the block's data, where each block is made of a header providing metadata, followed by a lengthy list of transactions, which together make up the majority of the blocks.

add_transaction(): Function adds the transaction to be mined in the next block, where the transaction records an occurrence, such as moving money from a sender's account to a beneficiary's account.

hash(): Function Provides a sha256 hash of the data in the block, whereas hash functions are a means to assure data integrity. After the data has been signed, someone may use a hash function as a check-sum to determine if it has been altered. It also functions as a way to confirm identification.

mine(): Function loop until the nonce satisfies the network target, where the first miner to guess the complicated code for the most recent block properly communicates their work to the rest of the network's miners. The original miner is rewarded if additional miners can validate the original miner's code.

isvalid(): Function checks each block in the chain, where a block must fulfill the network target, and the previous block's hash must match the current block's previous hash for each block in the chain.

solve_conflict(): The consensus algorithm is used in this function. Suppose another node has a valid chain and is longer than we have. This approach looks at the chains of the other nodes in the network and replaces our chain if a longer chain is discovered.

Algorithm-1 Pseudocode of __init__()

```
Input
    block sequence in the chain: block_number
    SHA-256 hash of the previous block's header: previous_hash
    approximate creation time of the block: timestamp
    arbitrary number that miners change repeatedly to produce the target: nonce
    series of actions: transactions
Output
    nothing
```

Algorithm-2 Pseudocode of add_transaction()

```
Input
    sender's account: sender
    recipient's account: recipient
    amount of transferring money: amount
Output
    list[sender, recipient, amount]
```

Algorithm-3 Pseudocode of hash()

```
Input
    block data: block_number, previous hash, timestamp, nonce
Output
    sha256 hash of block data
```

Algorithm-4 Pseudocode of mine()

```
Input
    sha256 hash of block data, difficulty
While
    sha256 hash > difficulty:
        nonce =nonce +1
        obtain hash value again
End While
```

Algorithm-5 Pseudocode of isvalid()

```
Input
    Current block data, sha256 hash of current block data, difficulty (network target), hash_of_previous_block
For
    each block in the chain:
        If
            previous hash <> hash_of_previous_block Or sha256 hash > difficulty
                return false
        End If
End For
return true
```

register(): Function that enables users to create trading accounts.

login(): Function uses username and password to authenticate a user.

send_money(): Function ensures that the user has enough money to transmit.

Algorithm-6 Pseudocode of solve_conflict ()

```
Input
        network_nodes_chains, our_chain_length
new_chain= false
For
        each node in the network_nodes_chains:
                If
                        node_length>    our_chain_length    And
                        node_chain isvalid()
                                our_chain_length= node_length
                                    new_chain = node_chain
                End If
End For
If
        new_chain<> false
                our_chain=  new_chain
                return true
End If
return false
```

Algorithm-7 Pseudocode of register()

```
Input
        username, name, email
get all values of the participants' usernames
check= false
If
        username Not in participants' usernames
                check= true
End If
If
        Check= true
                password = sha256 (password)
                add [username, name, email, password]
                auto login
Else
        error message: user is already existing
End If
```

Algorithm-8 Pseudocode of login()

```
Input
        username, password
(password_user): access password value related to inserted username
If
        password_user Is Non
                danger message: user is not existing
Else
        If
                password Is Match password_user
                        login user
                message: you are login successfully
        Else
                        Danger message: invalid password
        End If
End If
```

Algorithm-9 Pseudocode of send_mony()

```
Input
        sender, recipient, amount
(sender_balance): access to the amount sender possesses
If
        amount Is float
                error message: transaction is invalid
End If
If
        amount> sender_balance
                error message: inadequate Funding
Else If
        sender= recipient Or amount<=0
                error message: transaction is invalid
Else if
        Recipient's username Not in participants' usernames
                error message: user not exist
End If
End If
End If
(Recipient_balance): access to the amount Recipient possesses
sender_balance= - amount
Recipient_balance= + amount
```

IV. Results and Discussion

The outcomes of running the Python functions mentioned previously will be presented and examined in this section, and the variations will be discussed when supplying more than one state to each function will be discussed. We'll present these findings using visual studio code, the postman platform, and the google chrome browser. Through the use of Flask HTTP methods: get and post requests, we can communicate with our API to mine blocks, process payments, and observe the chain. As seen in Fig. 1, the chain has three blocks, each addressed by the hash_ value, block_ number, and previous_ hash. These blocks were only mined after the accuracy of their transactions had been confirmed. It is also noted that each block's hash_ value field equals the previous_ hash of its next block, where the proof-of-work consensus technique was used to confirm that the current block is valid as illustrated in Algorithm-5, that depended on network target value and previous_ hash field before being added to the chain and made publicly available. Where there are other consensus mechanisms, such as proof-of-stake, which chooses validators based on how many coins they own proof-of-authority, that facilitates relatively quick transactions using an identity-based consensus method, proof-of-elapsed-time allows the node with the lowest wait time will awaken first and win the block, which is permitted to add a new block to the blockchain, etc. Every node adopts the longest chain rule in the network to reach a consensus on the valid structure of the blockchain to resolve conflicts. Let's say that one node has more processing power than others, allowing it to create blocks faster. As a result, the unit with the longest length must be accepted as the valid version of the blockchain. For instance, node port 5000 has a chain of seven blocks, whereas node port 5001 has the longest chain in the network, containing nine blocks, as shown in Figs. 2 and 3, respectively. The solve_ conflict function eliminates these conflicts, as seen in Fig. 4, where the longest chain is adopted. Fig.5 illustrates the replacement of the node port (5000) chain with the longest chain.

The sending or receiving data from one address to another on the blockchain requires the other account's address. It alludes to a particular point on the network where crypto assets can be transmitted. To establish a new account, you must first sign up with your personal information (valid email, strong password, etc.). And according to algorithm 7, after confirming whether the user has no account, the password is transmitted to a safe format using SHA 256. A hash is a mathematical approach used in cryptography to transform arbitrary-length data into a fixed-length string, where the hash can be calculated quickly and easily. Still, it is a one-way method that makes it challenging or impossible to recreate the original input from scratch if only the hash value is known. The server receives a request for authentication and a payload that includes a username and password. When a user wants to sign in, the password given and the password saved are compared to the username in the table. The user can access the service after finding a match, as indicated by algorithm 8. There are sophisticated authentication techniques that boost security. For instance, Multi-factor Authentication (MFA), the approach used in the third related paper of this study, is an authentication technique that requires the user's provision of two or more verification elements to access a resource, such as an application or an online account. A robust identity must include in MFA as a fundamental element instead of merely

requesting a login and password. The balance verification is based on prior transaction linkages. For example, to transmit 10 bitcoins to someone, a transaction request must be produced that includes connections to prior incoming transactions totaling at least 10 bitcoins as shown in algorithm 9. These are known as "inputs." Nodes in the network validate the amount and guarantee that it has yet to be spent. In reality, if you link to inputs in a transaction, they are ruled invalid for any subsequent transaction. To limit the spread of transactions, as the algorithm proposed in the fourth related work submitted to avoid double-spend attacks, the previously spent coins must be invalid when all the subsequent payments are sent.

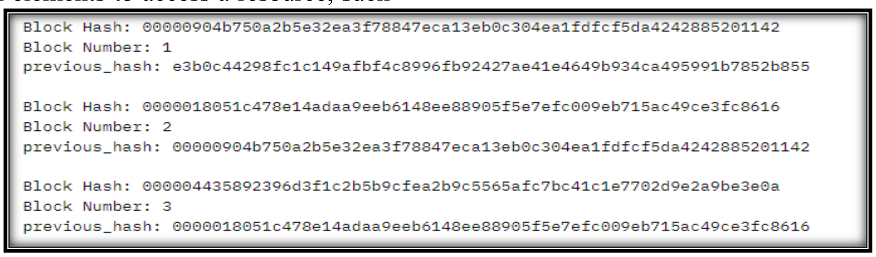

Fig. 1. Three blocks make up the chain.

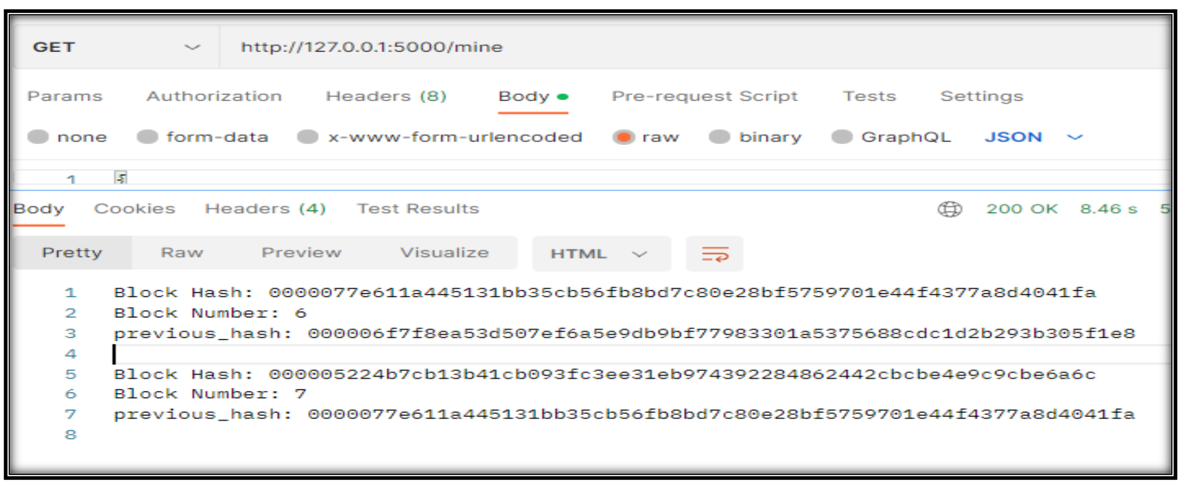

Fig. 2. Chain of node port (5000)

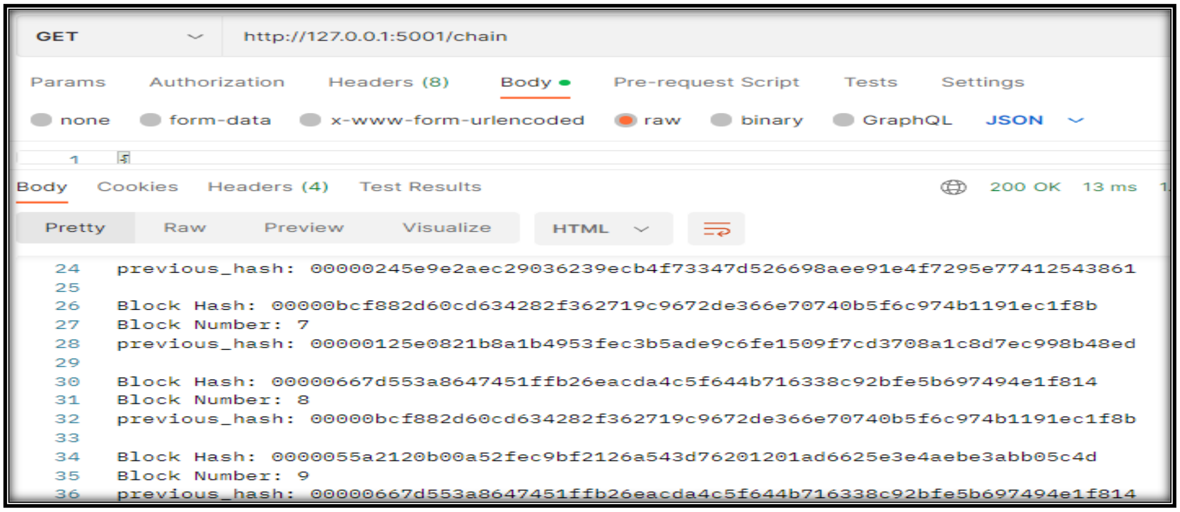

Fig. 3. Chain of node port (5001)

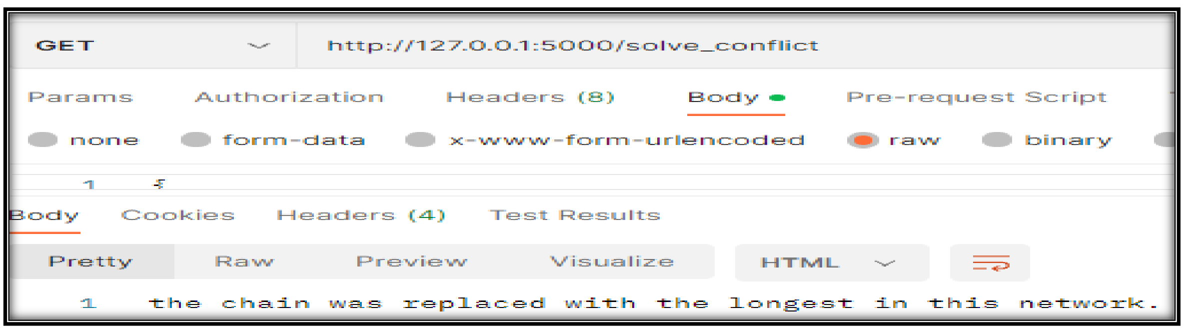

Fig. 4. Call up solve_ conflict function.

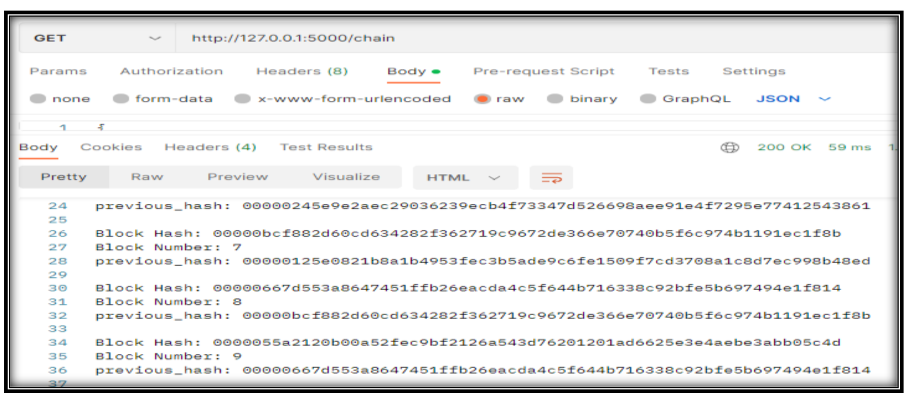

Fig. 5. The replacement with the longest chain

V. CONCLUSION AND FUTURE WORKS

Data movement between peers in a blockchain necessitates a series of verifications, particularly when the data is a cryptocurrency, to safeguard them against scammers who would do everything to obtain other users' coins. The other peers must validate each ongoing transaction from one peer before being added to their shared ledger. In this instance, a series of verification algorithms were submitted by our study. This includes a one-way hash method, conflict solving via the longest chain, prior hash to confirm that nothing has been manipulated when new blocks are added, balance verification, etc. As for future works, the verification procedure is better dependent on specific conditions such as the type of blockchain and general requirements. For instance, implement proof-of-authority consensus, where their identities are depended upon to add new blocks when the working environment is a private blockchain with a small number of participants, as opposed to using complicated computations, or enhancing the security of users' accounts, for as adding a one-time password that is generated specifically for each authentication occurrence.

REFERENCES

[1] A. M. Antonopoulos, Mastering Bitcoin: Programming the open blockchain: " O'Reilly Media, Inc.", 2017.

[2] I. Bashir, Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, ; distributed Ledger: Packt Publishing, 2018.

[3] M. Yano, C. Dai, K. Masuda, and Y. Kishimoto, Blockchain and Crypto Currency: Building a High Quality Marketplace for Crypt Data: Springer Nature, 2020.

[4] I. Bashir, Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more: Packt Publishing Ltd, 2020.

[5] M. Campbell-Verduyn, Bitcoin and beyond: Cryptocurrencies, blockchains and global governance: Taylor & Francis, 2017.

[6] V. Vallois and F. A. Guenane, "Bitcoin transaction: From the creation to validation, a protocol overview," in 2017 1st Cyber Security in Networking Conference (CSNet), 2017, pp. 1-7.

[7] M. Yaseen, M. Bahari, and O. A. Hammood, "Blockchain technology applications, concerns and recommendations for public sector," Mesopotamian Journal of Computer Science, vol. 2021, pp. 1-6, 2021.

[8] K. A. Taher, T. Nahar, and S. A. Hossain, "Enhanced cryptocurrency security by time-based token multi-factor authentication algorithm," in 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), 2019, pp. 308-312.

[9] H. Lee, M. Shin, K. S. Kim, Y. Kang, and J. Kim, "Recipient-oriented transaction for preventing double spending attacks in private blockchain," in 2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), 2018, pp. 1-2.

[10] S. Kim and G. C. Deka, Advanced applications of blockchain technology: Springer, 2020.