International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 02, 2020
ISSN: 1475-7192

A Framework for Detecting and Preventing Double Spent Data in Inter and Intra Block of Blockchain

[1]J. Vijayalakshmi, [2]A. Murugan

ABSTRACT--*Optimistically our global economy moves towards the digital eco-system. Everything is going to be paperless communication from investment to money transfer. The newest and most assuring digital payment sector is cryptocurrency. A cryptocurrency is a digital currency that uses cryptography as a medium of exchange like normal currencies. Bitcoin was the first decentralized cryptocurrency which has properties like supporting privacy, security, fast and cost-effective transactions, transparency and immutability transaction. Bitcoin is still dominant over 2000 cryptocurrencies. Double spending is the main issue faced by the Bitcoin network. Double spending is the method in which the same digital currency is spent multiple times by creating duplicate transactions. This issue needs to be resolved to make sure that it is not being abused, and that it maintains its value and trust. This paper has proposed several techniques for identifying and preventing double-spent transactions in both inter and intra block communication of the Bitcoin blockchain network.*

Keywords-- *Cryptocurrency, Bitcoin, Double spending, UTXO, Merkle*

I. INTRODUCTION

Digital currency or virtual currency is gaining popularity in the last couple of years due to the support of good properties like fraud-proof, identity theft management using public ledger, instant settlement and accessible from anywhere in the world. This digital currency provides digital transmission in a secured manner. This reduces the transaction complexity and prevents seeking of intermediaries like security brokers, financial agents, insurance agents, credit card companies, etc. [1]. Satoshi Nakamato introduced the first decentralized cryptocurrency called Bitcoin in the year 2009 for supporting virtual commodity which is treated as "Gold Standard" cryptocurrency. It is widely accepted by various retailers like Amazon, subway and Victoria secret [2]. Bitcoin is open-source software that supports multiple devices namely laptops, smartphones through the internet [3]. Bitcoin network allows the usage of bitcoins across anywhere in the world by any person at any time [4]. The structure of the Bitcoin network is depicted in Figure 1. This network comprises two parts namely blockchain data structure and Peer-to-Peer network. Blockchain data structure combines a series of blocks that are replicated and located in all participating nodes. This blockchain is required for verifying transactions that are installed in each peer [5].

[1] *Research Scholar, PG & Research Department of Computer Science,Dr. Ambedkar Government Arts College (Autonomous) Affiliated to University of Madras, Chennai, India,jeyamaha2002@gmail.com*

[2] *Associate Professor & Head, PG & Research Department of Computer Science,Dr. Ambedkar Government Arts College (Autonomous) Affiliated to University of Madras, Chennai, India, amurugan1972@gmail.com*

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

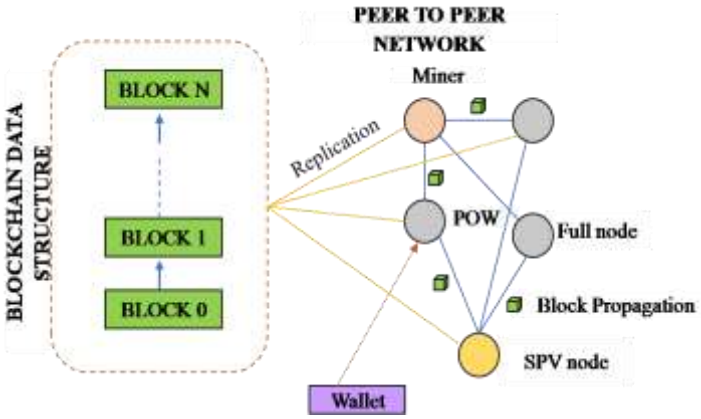


Figure 1: Overview of the Bitcoin network

Peer-to-Peer network helps to achieve file distribution in time based on considering basic parameters like file size, server count, receiving node count and capacities of participating nodes for uploading and downloading. The information propagated in the Bitcoin network is of two types namely block distribution and transaction distribution. If a node receives a block or transaction it verifies whether it is valid or not. If it is valid, the node relays it to other nodes. Before performing the validation of transaction or block, the Full node has to download and validate all the blocks and transactions of the blockchain ledger.

The node which downloads only headers and transactions alone and performs validation are called Simplified Payment Verification (SPV) nodes A wallet is a program which stores private and public keys which enable users to send and receive digital currency and monitor their balance. The client uses this wallet for doing transactions [5]. The miner who is a part of bitcoin network node gathers the transaction already propagated by users and organizes them into blocks which are then added to the blockchain. Each miner has to finish POW to add his proposed new block as the next block of the current blockchain and broadcasts the new block over the Bitcoin network.

II. DOUBLE SPEND TRANSACTIONS

According to bitcoin website [6] double spend is defined as

"***A transaction that*** uses ***the same input as an already broadcast transaction. The attempt of duplication, deceit or conversion, will be*** adjudicated ***when only one of the transactions is recorded in the blockchain***."

Satoshi Nakamoto [8] describes a solution for the problem of double-spending of bitcoin through network timestamps, a chain of hash-based proof of work and also specifies the probability rate that the attacker used to create double-spend data in a Peer-to-Peer network. Network timestamps proved that the data exist at a specific time. A previous timestamp is included in each timestamp forming a chain of hash as proof. Proof of work provided the historic record of transactions that is impossible to change by an attacker. The author said the probability rates that the attacker ever catches the honest chain are

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

$$1-\sum_{k=0}^{z}\frac{\lambda^k e^{-\lambda}}{k!}\left(1-(q/p)^{(z-k)}\right)$$

Where p denotes the probability of an honest node that finds the next block. q denotes the probability that the attacker finds the next block. q_z denotes the probability rate that the attacker ever catches up the z blocks behind.

$$q_z=\begin{cases}1 & if\ p\le q\\ (q/p)^z & if\ p>q\end{cases}$$

Xingjie et al. [9] proposed a method for detecting the double spending of bitcoin using decentralized non equivocation contracts. Here the sender locks few coins in a deposit when he initiates a transaction with the receiver. If the sender tries to double-spend an already spent coin then the calculative assertion will reveal his bitcoin credentials for the deposit and hence he lost his deposited coins. The fair deposit ensures that the sender will be penalized by the loss of deposit coins suppose if he tries to do double-spend and it will be compensated for the receiver's loss.

Karameet al. [10] denoted the accountability issues created by the misbehaviour of Bitcoin network. They said double-spending is a major problem in digital cash schemes where the same digital coin can be spent more than once because the digital files can be easily replicated. For that, they propose a new lightweight countermeasure which enables the detection of double-spending attacks on fast payments. They proposed a solution based on accountability and privacy definition for bitcoin. They measure and analyze the time required for transaction confirmation in Bitcoin through shifted geometric distribution. They also analyzed the privacy and accountability properties of Bitcoin.

III. DOUBLE SPEND DETECTION

A transaction is a process of transferring the ownership of bitcoins from one user to another user. Each transaction may consist of one or more input along with one or more output. The Spending of a previous transaction repository is called as input and transferring the repository to a new address is called as output [5]. Transaction performs the electronic mode of bitcoin exchange among the network peers. Each peer is holding bitcoin addresses which are used in every transaction process [11]. The unspent output from bitcoin transaction is the Unspent Transaction Output (UTXO). In a transaction, consumed UTXO are called transaction inputs and produced UTXO are called transaction outputs. The valid transactions are accepted and placed in the public ledger called blockchain. [12]. The following Figure 2 shows the process of transaction creation and addition into blocks of the blockchain. [7]

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

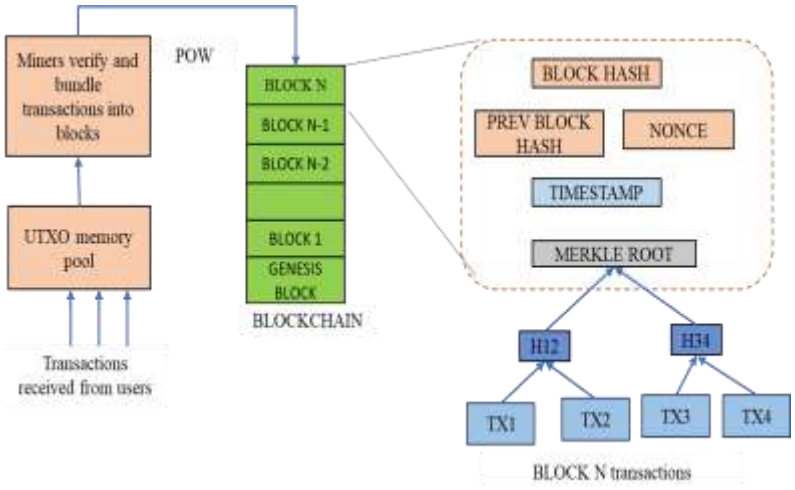


Figure 2: Transaction creation and addition in blocks

When a new transaction arrives, it is first entered into the UTXO memory pool. Miners in the Bitcoin network take the newly arrived transactions from the UTXO memory pool; perform verification and validation process using POW. Miners bundles the transaction into blocks which are then stored into the blockchain. The ownership of each bitcoin can be tracked via traversing the blockchain. Tampering of data is not possible, because the data might change the block hash. Miners add a new block to the blockchain by the following the set of process namely

- check the validity of a block by collecting the block details
- determining the valid hash value which is less than the target value of a block
- adding the new block in the local blockchain
- broadcast the solution

If the solution is correct then the miners update their local copy of the blockchain else discard the block and do validation for next block arrival. Blockchain forks arise due to factors like distributed block validation checking and network latency. Distributed block validation checking may determine the two possible solutions at the same time which produce fork. Similarly, network latency causes delay in verification of block which in turn creates a fork. The presence of blockchain forks in Bitcoin may create a double-spend attack easily. The following Figure 3 shows the double-spend attack generation due to race attack in the Bitcoin network.

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

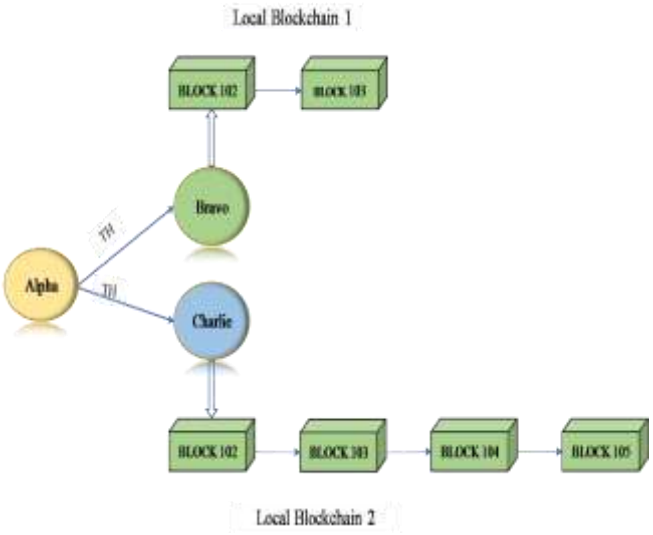


Figure 3: Creationof double-spend attack due to race attack

A race attack happens when a sender tries to spend the bitcoin at the same time in rapid succession. In the above process, the sender Alpha tries to spend the same BTC which specifies along with Transaction Hash (TH) value to the receiver Bravo and Charlie at a rapid sequence simultaneously. Here Bravo is the original recipient whereas Charlie is Alpha's friend. Unfortunately, Charlie's transaction has a long chain and Bravo's transaction chain is small. According to blockchain rule, the longest chain is adopted as current blockchain inclusion hence Bravo's transaction chain is rejected and Charlie's transaction chain is accepted which leads to the financial loss for Bravo and also makes the blockchain to an inconsistent state.

If this type of situation arises then how the Bitcoin network will handle this double-spent transaction arrival within the block (intra-block) and with multiple blocks (inter-block) communication are going to be discussed here in the following sections. First, the paper is going to discuss the double-spend detection at intra-block level then it discusses the double-spend detection at inter-block level using Cognizant Merkle and B-tree indexing. Next this paper is going to discuss double-spend prevention in intra-block level communication. Finally, a framework is constructed for detecting and preventing double-spend attacks in the Bitcoin network.

IV. DOUBLE SPEND DETECTION IN INTRA BLOCK LEVEL

The process of sending the same BTC to more than one receiver in a rapid sequence is called double-spending. This is often generated due to race attack. This attack normally arises due to the time delay of confirmation or the seller does not wait for confirmation. This attack happens mostly in fast payment scenarios like fast food payments, online services, vending machine payments, ATM withdrawals, etc [11]. The current Bitcoin system follows the method of the transaction included in the following way which is depicted in Figure 4.

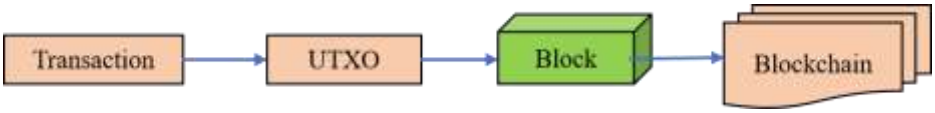


Figure 4: Bitcoin transaction inclusion process

Generally, all the incoming transactions along with their input and output are stored in the Unspent Transaction Output (UTXO) pool. Miners retrieve the transactions from the UTXO pool perform the validation by checking whether the input is available in the UTXO pool then they considered that it is not used by other transactions and hence they add the transaction into the upcoming block without validating whether the transaction used that output already or not. This might lead to the problem of double-spending since it is not validating fully. The proposed method of *Dual Payout based on Lost Agreement Amount* (DPL2A) [16] which is depicted in Figure 5 had a lot of pools for verifying whether the transaction is already spent or not based on UTXO pool.

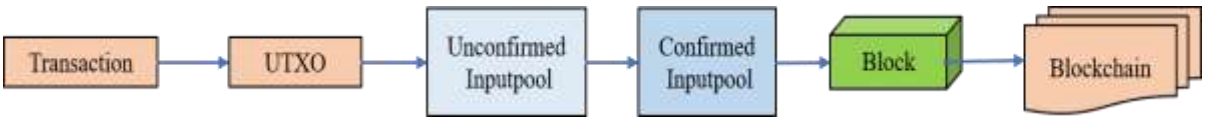


Figure 5: Dual Payout based on Lost Agreement Amount (DPL2A) architecture

Figure 6 specifies the transaction verification and validation based on DPL2A process, the transaction which enters the Peer-to-Peer network will enter into the *unspent transaction output* (UTXO) pool which was taken as input and it is moved to *unconfirmedinputpool* along with certain additional attributes for determining double-spend. The original transaction is identified from the *unconfirmedinputpool* by comparing the values of *unconfirmedinputpool* values with the *senderutxopool* and *senderstxopool*. If the attributes of these table like *sendername, transactionhash, outputindex*, transfer *value* and *time* match then those corresponding other attributes like *usedflag* and *usedcounter* part of both *unconfirmed inputpool* and *senderutxopool* and *senderstxopool* is updated.

Based on the status of *usedflag* and *usedcounter* properties of *unconfirmedinputpool* the double spended data was identified and stored into the *doublespend* table. Compare to standard bitcoin transaction verification this method DPL2A[16] additionally performs verification of transaction data along with various pools like *senderutxopool, senderstxopool* and *unconfirmed inputpool* for detecting the double-spend data. This method would detect the number of times the input reused and helps to reduce the miner's task difficulty. This method will identify the double-spent data successfully before it gets included in the block i.e., intra-block communication level. Next, we are going to detect the double-spent data in inter-block level communication.

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

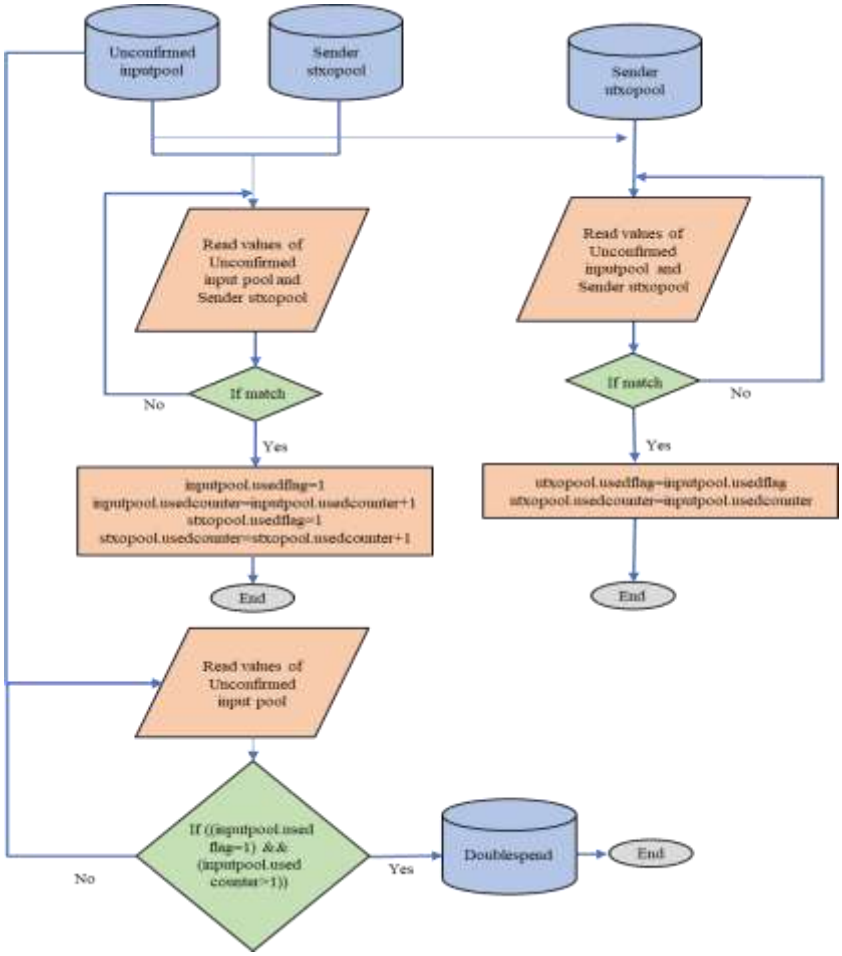


Figure 6: Dual Payout based on Lost Agreement Amount (DPL2A) process

V. DOUBLE SPEND DETECTION IN INTER BLOCK LEVEL

For multiple block-level identifications of double-spend data, the Merkle tree was used because this provide the digital fingerprint of all transactions inside the particular block. The Merkle trees are used to verify the transaction existence. These trees mainly use a Bitcoin light client protocol called simplified payment verification (SPV) [8]. SPV nodes are lightweight nodes where only the block headers and transactions alone are downloaded for transaction verification. SPV node helps to link the bitcoin transaction to the appropriate block and verify the transaction inclusion within that block. The path between transaction and block, block and blockchain gives the verification of transaction presence within the blockchain or not [13].

The proposed method of Cognizant Merkle[17] tree structure provided fast determination of transaction existence. This system constructs the Merkle tree based on the principle of HAT trie to achieve high speed compared to Bitcoin Merkle. HAT trie [14] comes in two forms namely hybrid and pure HAT trie. These forms are constructed based on the technique of bucket management and splitting. The pure HAT trie employs the Burst-trie technique[19] to achieve faster communication but it requires more space for application. The hybrid HAT trie applies the B-trie splitting algorithm which reduces the buckets and also cost and allowing several pointers for a bucket which gives faster access support which is shown in Figure 7.

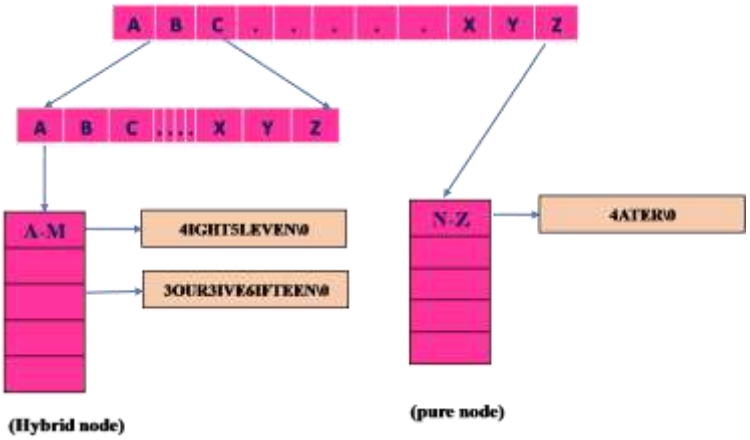


Figure 7: HAT trie with hybrid HAT-trie node representation for values EIGHT, ELEVEN, FOUR, FIVE, FIFTEEN and WATER

HAT tries are arranged and searched from top-down fashion which is a contrast to the Bitcoin Merkle tree where the tree is constructed from bottom-up fashion. When you store the element in a bucket, it does not allow duplicate values hence when the bucket gets full it is burst based on the type of HAT trie. For bursting [19] a suitable split point is chosen based on even distribution. To identify the string presence, the string is accessed by collecting all leading character occurrences and whose occurrence counters are traversed in a linguist order to determine the string movement. This Merkle tree is constructed based on the array of n+1 word length pointers. This bucket is used for storing base-64 encoded transactions. Base-64 encoding [15] is selected for encoding the transaction, because it can support all character types like ASCII, UTF8, UTF16 which can be easily transported to any systems in the network without modification or data loss. Figure 8 shows the Merkle root determination based on Cognizant Merkle structure [17]. Here the Merkle root is constructed based on applying Base 64 and Fast bitwise hash technique for the transaction.

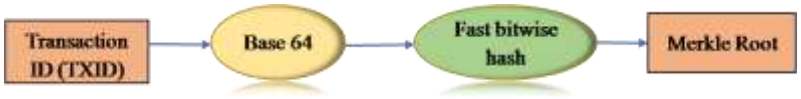
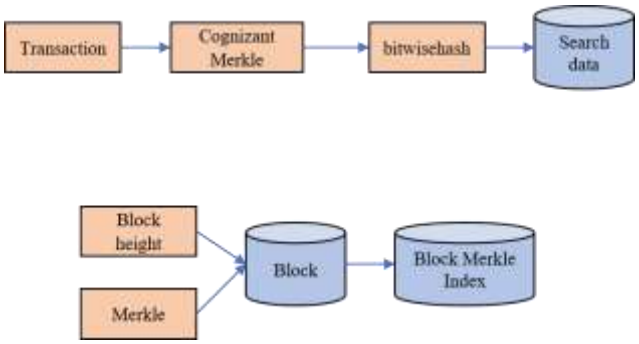


Figure 8: Cognizant Merkle construction



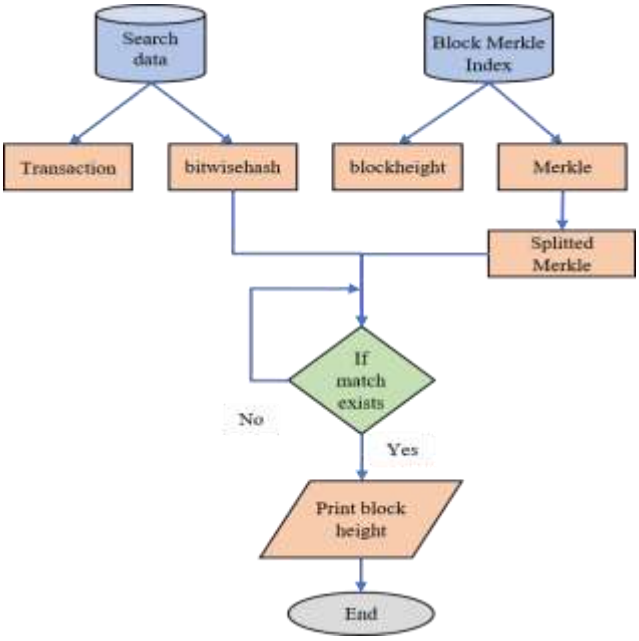


Figure 9: Multi-Block Double spend Transaction Detection (MBDTD) architecture

Double spending is a serious security issue in the Bitcoin network where any node can make faulty transactions by creating offline chains that lead to double-spent transactions in multiple blocks of blockchain. This degrades the peer-to-peer network data validation and integration. The entire blockchain network will fail in case of double-spent transactions in multiple blocks because these transactions cannot be removed or deleted once it gets added to multiple blocks. Blockchain properties like immutable, irreversible and tamper-proof will all fail due to this security issue. The process of detecting and analyzing the original transaction from multi-spent transactions in multiple blocks of blockchain is very tedious.

For this, the following architecture which is shown in Figure 9, *Multi-Block Double Spend Transaction Detection* (MBDTD) [20] will facilitate the detection of double-spending transactions in multiple blocks easily. This will also act as a prevention of double-spending transactions in multiple blocks in the future. Here the concepts of B-tree indexing and cryptographic hashing algorithm were used to find the detection of double-spending transactions in several blocks easily.

The indexing, *blockmerkleindex* was constructed based on B-tree indexing with the following set of values of *blockheight* and *blockmerkle* values. B-tree indexing gives faster access compared to normal access. This indexing will store the *blockheight* values in ascending order. If a transaction needs to check for its presence, it is first applied to the Cognizant Merkle for constructing temporary *Merkle* value. This temporary Merkle value was searched with all Merkle values of blockchain using *blockmerkleindex*. If there was a match then its corresponding block height values were retrieved which indicates that the particular transaction was available in that block.

VI. DOUBLE SPEND PREVENTION IN INTRA BLOCK LEVEL

To prevent the occurrence of double-spent data in intra-block level communication, first identify the list of double-spent transactions in the current block using Dual Payout based on Lost Agreement Amount (DPL2A) architecture with the help of *unconfirmedinput pool* list. The proposed architecture of *Authentic Contract Recognition based on Trans_UTXO_Input* (ACRT) [18] architecture which is depicted in Figure 10 will identify the original transaction from the set of double-spent transactions which was added to the *confirmedinputpool* which in turn added to the block of the blockchain.

Here the process of double-spend identification and prevention is carried out based on four transactors namely Alpha, Bravo, Charlie, and Delta. This framework consists of main pools like *transaction, block, blockchain* and some additional pools like *inputpool, outputpool, utxopool, alphastxopool, bravostxopool, charliestxopool,* deltastxopool, then additional pools like *alphautxopool, bravoutxopool, charlieutxopool, deltautxopool, alphapool, bravopool, charliepool, deltapool, unconfirmedinputpool, doublespend pool, confirmedinputpool, transhash pool, dspart1sol pool, dspart2sol* pool. DPL2A architecture identifies the set of double-spent transactions initiated by the same sender based on same transactionhash, same amount and with different sets of recipients.

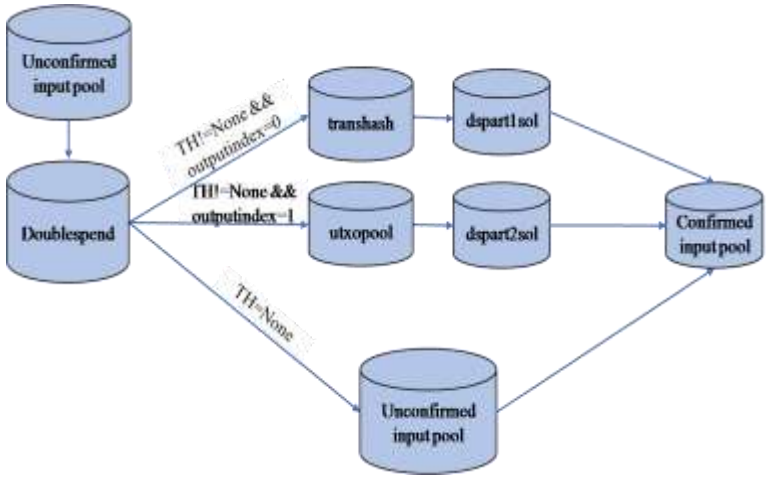


Figure 10: Authentic Contract Recognition based on Trans_UTXO_Input (ACRT) architecture

Consider the following double-spent transaction list in Figure 11. Here the transaction list transaction 1, transaction 2 and transaction 3 all having the same *sendername, transactionhash, outputindex,* value and time except *recipientname* field. These are said to be double-spent transactions that are stored in double-spend table. To identify the original transactions from this double-spent list, the pool like *transhash, unconfirmedinputpool* and *utxopool* values are compared [18].

Assume whenever the sender initiates transaction, initially it sets the recipient field as None. If the recipient uses this input for other transaction it retains this recipient field as None. Incase if a hacker tries to use this input for other transactions, he may not receive the amount because those who are having None as the recipient only can able to send and receive. Based on this principle the original recipient was identified by comparing multiple pools like *utxopool, unconfirmedinputpool* and *transhash* pool values.

transaction1:

sendername: <__main__.Blockchain instance at 0xb63099cc>

transactionhash: b4418f0b8d4ec73ec3aa2921ab72faccaad0d509f14c0f55019fda5599e6e250

outputindex: 0

value: 30.0

time1554706418.77

usedflag: 1

usedcounter: 6

recipientname: <__main__.Blockchain instance at 0xb6309a0c>

transaction2:

sendername: <__main__.Blockchain instance at 0xb63099cc>

transactionhash: b4418f0b8d4ec73ec3aa2921ab72faccaad0d509f14c0f55019fda5599e6e250

outputindex: 0

value: 30.0

time1554706418.77

usedflag: 1

usedcounter: 6

recipientname: None

transaction3:

sendername: <__main__.Blockchain instance at 0xb63099cc>

transactionhash: b4418f0b8d4ec73ec3aa2921ab72faccaad0d509f14c0f55019fda5599e6e250

outputindex: 0

value: 30.0

time1554706418.77

usedflag: 1

usedcounter: 6

recipientname: <__main__.Blockchain instance at 0xb630972c>

Figure 11: double-spent transaction list

VII. ARCHITECTURAL FRAMEWORK FOR BITCOIN DOUBLE SPENDING DETECTION AND PREVENTION

The following Figure 12 gives the architectural ***Framework for Double Spend Detection and Prevention* (F2DP)** for double-spend detection and prevention of the Bitcoin network. Here the sender and receiver are performing the transaction process through the wallet. A sender is sending his coins through his *wallet* to the receiver through the *Peer-to-Peer* network. *Peer-to-Peer* network, in turn, communicates with the *peer discovery* of Bitcoin core architecture. The *peers* database holds the list of peer nodes in the Bitcoin network and the

connection manager is responsible for handling transaction insertion and block creation of the blockchain. On transaction creation, the input is taken from the *UTXO* pool and performs transaction initiation.

For performing double-spend detection and prevention, the *miners* choose the input from the *UTXO* list. The unspent input from the *UTXO* list is moved to an *unconfirmed inputpool* list where the intra-block double-spent data is detected using *DPL2A* architecture. The *doublespend* data are retrieved based on comparing the *unconfirmedinputpool* with the *senderutxopool* and *senderstxopool* values. Based on the status of *usedflag* and *usedcounter* attributes of *unconfirmedinputpool* the *doublespend* data are identified and it is moved to *doublespend* table. The original transaction is detected from *doublespend* table using *ACRT* architecture.The ACRT architecture identify the original transaction using *transactionhash*(TH) and *outputindex* field of *doublespend* table.

If TH!= “None” and outputindex=0 , then the transaction from the *doublespend* table is compared with *transhash* table to identify the original transaction which is stored in *dspart1sol* table. If TH!= ”None” and outputindex=1, then the transactions of *doublespend* and *utxopool* tables are compared to identify the original transaction which is stored in dspart2sol table. If TH=”None”, the *doublespend* table transaction is compared with unconfirmedinputpool to retrieve the original transaction. Finally all the transactions from dspart1sol, dspart2sol and original transactions are moved to the *confirmedinputpool* which in turn moved to the *mempool* maintained by the miners. After performing transaction validation, the verified transaction is moved to *blocks* where block Merkle is constructed using *Cognizant Merkle*. By combining *blockheight* and *Cognizant Merkle* the *blockmerkleindex* table is constructed.

For the searching list of double-spent transactions *tempmerkle(bitwisehash)* value is generated using Cognizant Merkle. The *searchdata* table stores the double-spent transaction along with *tempmerkle* value. For identifying the double-spent data along multiple blocks (inter blocks), the *MBDTD* architecture is used. *MBDTD* architecture compares the *tempmerkle* value with the block Merkle value (*CognizantMerkle*) of each block. If it matches, the corresponding *blockheight* was returned as output which denotes the doube-spent transaction is available in that block number.

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

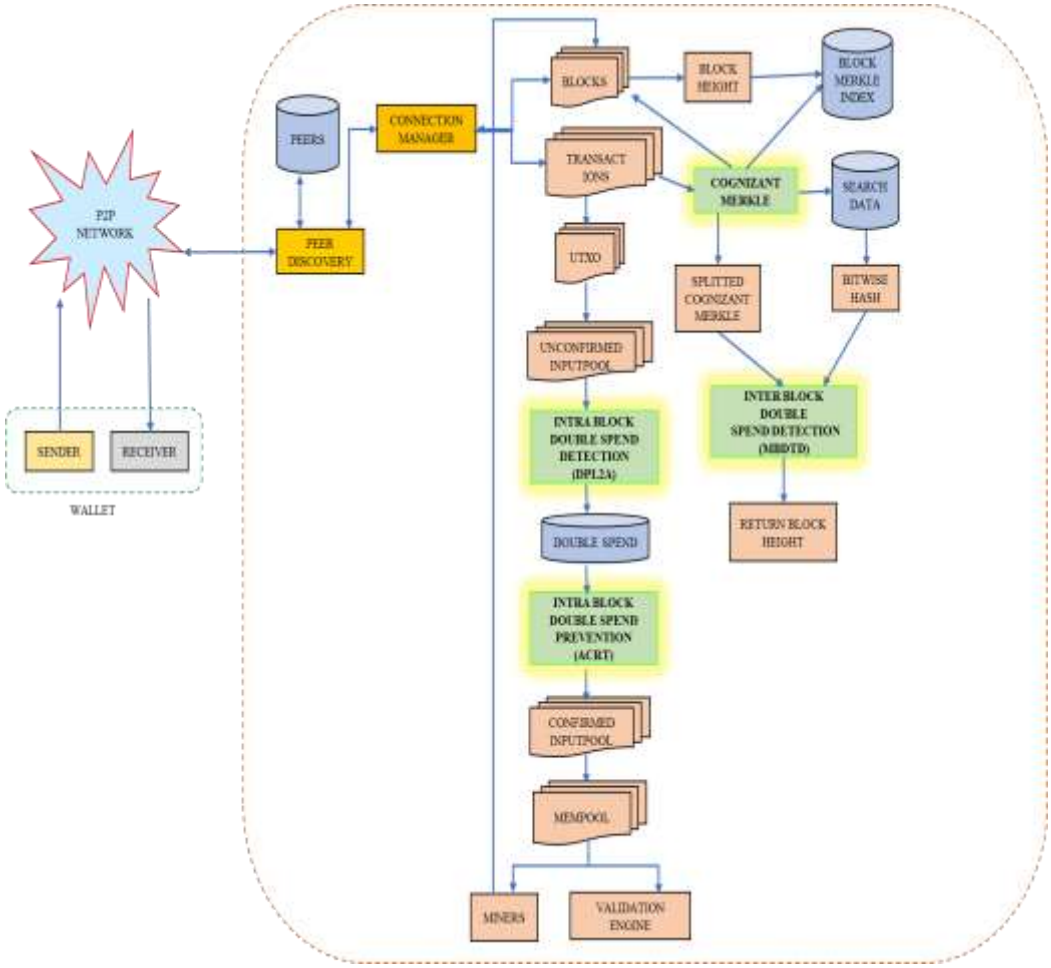


Figure 12: Framework for Double Spend Detection and Prevention (F2DP)

VIII. CONCLUSION AND FUTURE SCOPE

This paper has introduced a modified framework of existing Bitcoin core architecture [18] for detecting and preventing the double-spent transaction data in both inter and intra-block communication levels. First, the paper has discussed the Bitcoin network overview then it analyze the existing solution of double-spending problem provided by various authors. Next, it discusses the various architectures like DPL2A, MBDTD and ACRT for handling double-spend detection and prevention in inter and intra-block communication. The double-spent data detection in intra-block communication is identified through DPL2A architecture. For double-spent transaction identification in multiple blocks, MBDTD architecture were used along with B-tree index and Cognizant Merkle. The prevention of double-spent data in intra-block communication is carried out based on ACRT architecture. Finally, this paper has introduced a new framework F2DP for detecting and preventing the double-spent transaction by customizing the Bitcoin blockchain. In Future, solution for double-spend data done by selfish miners like 51%attack and vector 76 attacks can be implemented. Mitigation of scalability issue is also considered with the help of Cognizant Merkle.

Received: 02 Dec 2019 | Revised: 24 Jan 2020 | Accepted: 16 Feb 2020

REFERENCES

1. Vasek Marie. "The age of cryptocurrency How Bitcoin and Digital Money are Challenging the Global Economic Order", *Science*, Vol. No. 348(6241), PP 1308-1309, 2015.
2. Vijayalakshmi, J., and A. Murugan. "Crypto coin overview of basic transaction." *International Journal of Applied Research on Information Technology and Computing* Vol.No.8(2), PP 113-120, 2017.
3. Antonopoulos M. Andreas "Mastering Bitcoin: unlocking digital cryptocurrencies" *O'Reilly Media*, 2014.
4. Oliveira Samuel, Filipe Soares, Guilherme Flach, Marcelo Johann, and Ricardo Reis. "Building a bitcoin miner on an FPGA." In *South Symposium on Microelectronics*, 2012.
5. Shahsavari Yahya, Kaiwen Zhang, and Chamseddine Talhi. "Performance modeling and analysis of the bitcoin inventory protocol." In *International Conference on Decentralized Applications and Infrastructures* (DAPPCON), IEEE, 2019.
6. Definition of double-spend, "https://bitcoin.org/en/glossary/double-spend", [online accessed 24-oct-2019]
7. Conti Mauro, E. Sandeep Kumar, Chhagan Lal, and Sushmita Ruj. "A survey on security and privacy issues of bitcoin." *IEEE Communications Surveys & Tutorials*, Vol.No.4(20), PP 3416-3452, 2018.
8. Nakamoto, Satoshi, "Bitcoin: A peer-to-peer electronic cash system.", "https://bitcoin.org/bitcoin.pdf", 2008.
9. Yu Xingjie, Michael Thang Shiwen, Yingjiu Li, and Robert Deng Huijie. "Fair deposits against double-spending for bitcoin transactions." In *IEEE Conference on Dependable and Secure Computing*, PP 44-51, 2017.
10. Karame Ghassan O, Elli Androulaki, Marc Roeschlin, Arthur Gervais, and SrdjanČapkun. "Misbehavior in bitcoin: A study of double-spending and accountability." *ACM Transactions on Information and System Security*, Vol. No. 1(18),2015.
11. Karame Ghassan, Elli Androulaki, and SrdjanCapkun. "Two Bitcoins at the Price of One? Double-Spending Attacks on Fast Payments in Bitcoin." *IACR Cryptology*, Vol.No.248, 2012.
12. Ouaddah Aafaf, Anas AbouElkalam, and Abdellah Ait Ouahman. "Towards a novel privacy-preserving access control model based on blockchain technology in IoT." In *Europe and MENA Cooperation Advances in Information and Communication Technologies*, Springer, PP 523-533, 2017.
13. Knecht Markus, Willi Meier, and Carlo U. Nicola. "A space-and time-efficient Implementation of the Merkle Tree Traversal Algorithm." *arXiv preprint*, Vol.No.4081(1409), 2014.
14. Askitis Nikolas, and Ranjan Sinha. "HAT-trie: a cache-conscious trie-based data structure for strings." In *Proceedings of the thirtieth Australasian conference on Computer science*, Australian Computer Society, Vol. No 62, PP 97-105, 2007.
15. Stackoverflow questions, "https://stackoverflow.com/questions/4070693/what-is-the-purpose-of-base-64-encoding-and-why-it-used-in-http-basic-authentica" [online access date: 13- feb-2018]
16. A. Murugan and J. Vijayalakshmi. "Discovering the Bitcoin Double Spend using Lost Agreement Amount." *International Journal of Recent Technology and Engineering*, Vol.No.8(3), PP 3764-3770,2019.

17. Vijayalakshmi, J. and Murugan A. "Revamp Perception of Bitcoin Using Cognizant Merkle." In *Emerging Research in Computing, Information, Communication and Applications*, Springer, Singapore, PP 141-150,2019.
18. A. Murugan and J. Vijayalakshmi. "Preventing the bitcoin Double Spend using Transaction Hash and Unspent Transaction Output." *International Journal of Recent Technology and Engineering*, Vol.No.8(3), PP 3771-3776,2019.
19. Heinz Steffen, Justin Zobel, and Hugh Williams." Burst-tries: a fast, efficient data structure for string keys", *ACM Transactions on Information Systems*, Vol. No. (2), PP 192-223, 2002.
20. Vijayalakshmi, J. and Murugan A. "Detecting Multi-Block Double Spent Transaction based On B-tree Indexing", International Journal of Scientific and Technology Research(communicated)