# RDV: Register, Deposit, Vote
# Secure and Decentralized Consensus Mechanism for Blockchain Networks

Siamak Solat

UPMC-CNRS, Sorbonne Universités, LIP6, UMR 7606, Paris, France

**Abstract.** A decentralized payment system is not secure if transactions are transferred directly between clients. In such a situation it is not possible to prevent a client from redeeming some coins twice in separate transactions that means a double-spending attack. Bitcoin uses a simple method to preventing this attack i.e. all transactions are published in a unique log (blockchain) [7,28]. This approach requires a global consensus on the blockchain that because of significant latency for transaction confirmation is vulnerable against double-spending. The solution is to accelerate confirmations. In this paper, we try to introduce an alternative for PoW because of all its major and significant security problems that lead to collapsing decentralization of the Bitcoin, while a full decentralized payment system is the main goal of Bitcoin idea. As the network is growing and becoming larger day-to-day, Bitcoin is approaching this risk (see Figure 3). The method we introduce is based on a distributed voting process: RDV: Register, Deposit, Vote.

## 1 Introduction

In Bitcoin a client requires 6 "confirmations" to be sure the transaction is not reversible (that means 6 blocks must be generated over a client's transaction) . It is interesting that choosing 6 blocks is arbitrary and it is not based on any analysis or probability of forks [7]. Also, Bitcoin using an ad-hoc approach tries to prevent deep forks i.e. including "hard coded blockchain prefixes" as checkpoints by default clients. Laurie [12] proves that using these check points demonstrates Bitcoin system is not a decentralized consensus method since these checkpoints are selected by a centralized method. A significant latency between two blocks and receiving by the rest of network increases possibility of forks. In the other side, a miner who controls a significant number of nodes in the network can increase probability of wining of their branch in a fork by transmission of their own blocks and rejecting the other ones [7]. there are proposals to reduce the latency in the Bitcoin network [3] [13] [14]. Bitcoin's original paper introduces an informal explanation for eventual concensus [1]. However, other researches shows in presence of some conditions like timely propagation channel and honesty of miners' majority in the network, eventually Bitcoin reaches a consensus [5] [15]. In case of temporary fork, it is possible for an attacker to make a double-spending attack [16] [17]. Because of latency in block generation and transaction

confirmation the scalability is another problem of current Bitcoin protocol. An attempt on this matter is introduced by [21] called as blockchain pipelining by adding "blockchain-like characteristics to the distributed DB". Another question is that who is eligible to vote for a transaction? In Bitcoin, the miner who can solve proof-of-work faster than others is eligible to vote, thus one votes and others have to accept it. This is not a fair voting process. Also, consider the fastest miner is an adversary. In RDV, the nodes who have registered are eligible to vote and almost everybody is able to register, thus almost everybody are permuted to participate in voting and as a result, RDV is more democratic than Bitcoin.

## 2 Motivation

Here we explain our motivation to replace PoW by RDV:

**Safety Threshold:** In PoW, safety threshold against attack depends on hashing power of adversary regardless of adversary's size (in number), but in RDV, it depends on "number" of adversarial nodes i.e. adversary's cartel size.

**Participant's Requirements to Participate:** In PoW, participant's guarantee and requirements to participate in transactions confirmation is external source i.e. *cpu and processors*, but in RDV, source is internal i.e. *deposited coins*. The type of source (internal vs. external) can reduce significantly the motivation of destroying the system by adversary.

**Possibility to Participate:** in PoW: possibility to participate in transaction confirmation is possible for powerful (or fast) nodes in hashing operation, but in RDV, it is possible for almost all nodes.

**Probability of Wining:** In PoW, probability of wining and receiving reward depends on how much you are fast in hashing operation, regardless of number of honest nodes in the network and your honesty, but in RDV, it depends on how many nodes in the network are honest and your honesty as well.

**Majority:** In PoW, if majority in the network are honest, but you are much faster, you likely win even if you are not honest, but in RDV, if majority in the network are honest and you are honest too, always you win, but if you are not, always you lose a part of your coins (as guarantee) that reduces significantly adversary's motivation.

**Decentralization of the Network:** Decentralization of the Bitcoin network is currently under the risk, since finding the PoW's answer and as a result block generation is very difficult. Thus, only the mining pools that own major hashing power with very fast processors are able to determine fate of transactions and network situation. In the other side, if a mining pool achieves more than 51 percentage of total hashing power of the network, according to 51% attack it is able to control the network. Currently an organization [25] owns more than 51% hashing power of the network and it means "we have to trust" this organization or any other similar mining pool with such this hashing power which means collapsing decentralization of the network. This means **we have nothing**, when the goal is having a decentralized payment system.

**Full Replication:** In Bitcoin network, if you intend to participate in mining process and receive related reward, you need to have the whole of blockchain (i.e. replica) to calculate the related hash to be able finding the correct nonce in PoW. Over time the blockchain size grows and thus only the nodes with enough resources to hold the data are able to be a participant. This property is another weakness which threatens decentralization of the network. We show our solution for this problem such that a participant does not need to be a "full node" (i.e. a node which holds all data.)

**Preventing Block-withholding:** One of main security problems in Bitcoin system is block-withholding that there are some solutions for this attack[26,27]. RDV can prevent block-withholding attack. Since in current Bitcoin network after finding correct answer of PoW and discovering a new block a mining pool the whole of blockchain will be replaced by propagating the new blockchain and all nodes in the network hold longest chain, thus a selfish miner keeps its new block private until in an appropriate opportunity publishes selfish blockchain in the entire network to receive more reward. But since in RDV idea a node receives a reward for participating in voting process if and only if the participant's vote is equal to voting process result, thus there is no opportunity for an attack such as block-withholding.

**Stronger Decentralization:** Since in RDV each block consists of only one transaction, so this leads to increasing number of blocks significantly. Comparing with Bitcoin system in which because of difficulty in solving PoW and mining process there is a significant latency in block generation which causes inserting several transactions in one block that decreases decentralization of the system importantly.

**Energy Consumption:** Bitcoin uses significant amount of energy. This leads to be problematic in long term. Currently, the Bitcoin average electricity consumption in half an hour is equal to US household average electricity consumption in one year.

## 3 Interior vs. Exterior resources:

In gneral security of a system must not be depended on only exterior resources. For example, in Bitcoin if a miner has access to a free electricity resource, then security of the system faces significant dangerous risk, since the cost of necessary electricity for mining process is not modifiable via inside of the system. On the other hand, if a miner has access to free electricity resources, he / she does not spend any penalty for his / her aggressive behavior like forking blockchain. In other words, forking blockchain has no cost for the adversary.

The security parameters must be modifiable via the system. However, an exterior resource can be very useful as a "complementary" parameter. As a result, exterior resources can be employed as a complementary, but the main resources must be constitutive entities of the system (ex. coins in a cryptocurrency network). However, we must design an appropriate **incentive-punitive system**

such that according to the Nash equilibrium diverging from the algorithm does not lead to a net profit [6] to achieve an ideal cryptocurrency.

## 4 RDV main steps

- *Register:* Each node for being able to vote for a transaction has to register, otherwise the node is "ordinary" which is only able to pay / receive a transaction and coins. All "registered" nodes are listed into a field inserted into the blockchain.
- *Deposit:* It means blocking some coins for being able to finish registration step. This amount is calculable regarding to the total amount of coins in the whole of network along with a coin value. The registration process is acceptable if and only if a part of the coins of "volunteer" node for participation in voting process is blocked in depository of the network such that the registered node has no access to this part of its coin as long as the node is a "voter" (i.e. a registered node).
- *Vote:* Each registered node is able to vote for a transaction either positive (i.e. 1) or negative (i.e. 0) In case the result of voting process is not equal to a voter's vote, then this node will lose a part of its blocked coins "for ever" as a penalty for preventing aggressive behavior. The amount of this penalty is also calculable like previous step. In the other side, if the result of voting process is equal to the voter's vote then for incentivization and motivation, this node receives some coins as a reward.

*Preventing Aggressive Behavior:* A user can vote for a transaction using several nodes (computers), but the user has to register to each of computer that means blocking a part of coins (ex. $cb$ coins) for every computer i.e. $cb \times n$, thus in case the voting result does not become equal to this user's vote, then he / she has to pay a significant of his / her coins as a penalty. This decreases significantly risk for voting "aggressively" several times. Also, in such a situation this user's physical address will be blocked forever that means the user is not able to register as a voter using this physical address. Thus, it causes some additional external cost (i.e. several computers) plus some internal cost (i.e. blocking some coins for registration to be authorized for voting process). See Figure 4. As a result, RDV avoids the Sybil attack better than PoW by more additional cost (both external and internal cost). In case an adversary uses a virtual machine, then if network monitors the switch MAC table, they will always be able to see two MAC addresses on the physical port and so they would know a second device is attached either with a repeater or virtual machine.

## 5 Blockchain Consensus Approaches

In this section we compare RDV approach with other major consensus protocols.

1. **proof-of-work:**
   The major example uses PoW for consensus of blockchain network is Bitcoin. A proof-of-work (PoW) is a cryptographic puzzle that is difficult to solve but easy to verify. Bitcoin network uses Hashcash [2] proof-of-work system for block generation such that a block is accepted by the network if miners perform proof-of-work properly and successfully. The difficulty of PoW is adjustable regarding to the hashing power of the network. A mining reward motivates the miners to raise use of their resource. A transaction confirmation by a mining pool takes around 10 minutes on average. Some other alt-coins like Litecoin [29] decrease this latency, however they reduce the security. In proof-of-work there is not any limitation on who can join the network as a voter. A miner is winner to receive related reward at random, proportional to its hashing power. PoW has a propensity for being centralized, since it is like a competition to collect the highest hashing power which is currently in hand of a small number of mining pools.
   **Sybil attack:** An attacker is able to try for achieving majority of nodes in the network under his/her control. In this case a client with high probability is connected only to the adversary nodes. It causes the following related security problems:
   According to [8] and [9] solving the general problem of consensus in a distributed network is impossible without further assumptions. For example Bitcoin system assumes that the rational behavior may be modeled [7]. Despite mentioning PoW detects the Sybil attack (or sockpuppet) vaguely [8] [7] in fact PoW only can avoid it if the attacker intends to do hashing operations with lower difficulty level. The Sybil attack means the attacker is able to swap up between several nodes under his control and this is possible in presence of proof-of-work. We explain how other problems are possible to occur even if we use proof-of-work in the network [10].

   (a) The first one is refusing to relay blocks that does not belong to the attacker.
   (b) If the attacker relays only himself blocks and transactions (in his / her mining pool) then everyone is in risk of double-spending attack.
   (c) If a node relays transactions with zero confirmation, then the attacker is able to filter certain transactions to make a double-spending attack.
   (d) If a node is connected to several nodes belonging to attacker and attacker would be able to monitor the transmissions of this node, then this attacker is able to prevent the "low latency encryption" of the Bitcoin network using a "timing" attack.

   As a result, we see that despite preventing hashing operation with lower difficulty level, however there are several other possible attacks in presence of PoW. In the other hand, an attacker mining pool is able to do a Sybil attack by adding a significant number of zero-power nodes in the network as a virtual miners that means the nodes do not have a power for hashing operation but also they work as some sensors to monitor the network and honest miners' behavior by participating in data dissemination [11] [4]. Some

other attempts for consensus in blockchain is introduced by [**?**] a multivalued Byzantine consensus algorithm tailored for consortium blockchains.
Confirming tx with lower amounts has lower reward. So, a miner usually chooses tx with higher amounts. As a result, micropayments need to wait too much to be confirmed.
For achieving a cost-effective PoW, we need to increase difficulty level. So, an honest miner with a single (even fast) processor is not able to find correct nonce and thus loses his motivation to participate. In the other hand, a rational organization with a large network of processors has a chance to find correct nonce. In such a situation, if this rational organization after finding the correct nonce decides to postpone a correct tx by not inserting tx in his block, or confirm an incorrect tx by inserting in his block, is there a penalty for this rational organization? Of course there is not. This shows lack of an effective incentive-punitive method.
Another problem is that processing / hashing power of miners is extremely varied. For example, mining 10 minutes with a 3GHz Pentium is equal to one hour mining with Palm Pilot [**?**].

2. **proof-of-stake:**
The major example uses proof-of-stake for consensus of blockchain network is Ppcoin [18]. In PoS for block validation a node is selected randomly, and proportionally to percentage / amount of its coins. PoS decreases latency and does not need huge external resources for hashing functionality like PoW. Tendermint [24] is a round based and DLS [23] based algorithm which is resilient up to 1/3 of adversaries. It has been designed based on section 4, algorithm 2 of [23] and employs a full-replication of public ledger to store history of transactions like Bitcoin protocol. Also it uses round robin method in which block validators are selected alternatively to confirm a new block [24].
3. **Mixed PoW/PoS:**
The major example uses Mixed PoW/PoS for consensus of blockchain network is Ethereum [19]. In fact, if there are a few coins in the network, it is easy for an adversary to buy the majority of coins and simply get control of the network. Thus, Ethereum team decided to start with PoW to get enough coins into the network and then switch to PoS.
4. **Federation:**
The major example uses Federated blockchain network is Stellar consensus protocol [20]. In a federated model there are several groups in which of them a node operates according to the group's rules set.
Another attempt in federated consensus is Ripple [22]. While Bitcoin uses mining operation for consensus, Ripple employs an iterative process.

### 5.1 RDV Algorithm Detailed Description

*line* 1: voter node starts an infinite while loop.
*line* 2: voter checks his *tx* box to know if there is a new *tx*.

**Algorithm 1** RDV algorithm

**Require:**
txBox[.] List if transactions a voter receives from the network.
voteBox[TX.Id][voterId] keeps votes values.
voterBox[.] keeps list of voters' ID.

**Ensure:**

**while** () **do**
  $TX \leftarrow$ ***CheckTxBox()***
  **if** $(TX \neq null)$ **then**
    $txBox[.] \leftarrow sortTX(TX.am, TX.ts, txBox[.])$
  **end if**
  $TX \leftarrow txBox[0]$
  **if** $(readTX(TX) = true)$ **then**
    $TX.Id.voterId \leftarrow 1$
  **else**
    $TX.Id.voterId \leftarrow 0$
  **end if**
  $voteBox[TX.Id][voterId] \leftarrow TX.Id.voterId$
  $broadcastVoter(voteBox[.][.])$
  $voterBox[.] \leftarrow updateVoterIds(voterBox[.])$ ▷ updating voters nodes
  $voteBox[.][.] \leftarrow updateVoteBox(voteBox[.][.])$ ▷ receiving votes from others
  **while** (ParticipateAllVoter(voterBox[.] $\neq$ true)) **do**
    $NotParticipate24[.] \leftarrow NotParticipate24()$
    **if** $(NotParticipate24[.] \neq null)$ **then**
      Remove voters id without participation in 24 hours ago
      Block related physical addresses for voting process for ever
      updateVoterBox(voterBox[.])
    **end if**
    **if** (ParticipateAllVoter(voterBox[.] $\neq$ true)) **then**
      $NewVoter \leftarrow checkNewVoter()$
      **if** $(NewVoter \neq null)$ **then**
        $voterBox[.] \leftarrow updateVoterIds(voterBox[.], NewVoter)$
        $voteBox[.][.] \leftarrow updateVoteBox(voteBox[.][.])$ ▷ new votes
      **end if**
    **end if**
  **end while**
  $votesResult[TX.Id, ones, zeros] \leftarrow countingVotes(voteBox[.][.], TX.Id)$
  **if** $(TX.Id.ones > TX.Id.zeros)$ **then**
    $B_i \leftarrow genBlock(B_i)$
    $blockchain \leftarrow insertBlock(B_i)$
    $broadcastAll(blockchain)$
  **end if**
  $NotParticipate24[.] \leftarrow null$
**end while**

*line* 3: If so, voter inserts and sorts his *tx* list.
*line* 6: voter selects a *tx* which has most priority point (i.e. txBox[0])
*line* 7: voter reads *tx* and verifies if it is done properly.
*line* 8: If so, voter votes for tx (i.e. 1).
*line* 10: otherwise, voter's vote is 0.
*line* 12: voter sets his voteBox[txID][voterID] to his vote value (i.e. 1 or 0).
*line* 13: voter then broadcasts his voteBox to all registered nodes.
*line* 14: voter updates list of voters to know who joint or left the network.
*line* 15: voter checks his vote box to know vot other votes.
*line* 16: voter starts a while loop and exits when all voters have participated.
*line* 17: voter using NotParticipate24() checks if there is a voter who have not participated in any voting process since 24 hours ago.
*line* 18: If there is: then:
*line* 19: related voter id is removed from voterBox.
*line* 20: and his physical address is blocked for ever.
*line* 21: voter updates his voterBox.
*line* 23: voter checks if all voters have participated.
*line* 24: If not, voter checks if there is a new joint voter.
*line* 26: If so, voter updates voterBox.
*line* 27: and then updates voteBox to gather new votes.
Then voter goes back to line 16.
When all voters participates in voting process, then voter breaks while loop and jumps to line 31.
*line* 31: voter starts to count the votes, including zeros (negative votes) and ones (positive votes).
*line* 32: If ones are greater than zeros, then:
*line* 33: create a new block including *tx*.
*line* 34: voter inserts new block into blockchain.
*line* 35: voter broadcasts blochchain to entire network, including registered and ordinary nodes.
*line* 37: voter cleans Notparticipate24[] list and then goes back to line 1.

### 5.2 Correctness of RDV Consensus:

We divided the nodes into two sets: ordinary and registered. Then we define $state_1$ in which we have set of registered nodes as follows:

RNset = $\{rn_1, rn_2, \ldots, rn_n\}$

If all of them participate in voting process, then we achieve consensus, otherwise we define a $state_1$ in which:

$state_1$: One of them does not participate in voting process within 24 hours, then it will be removed from RNset and his / her physical address will be blocked forever and his / her blocked coins will be deleted. (see algorithm **??**). As a result, *(n - 1) rn* will achieve a consensus.

$state_2$: Two of them do not participate in voting process within 24 hours, then these two *rn* will be removed from RNset and their physical address will

be blocked forever and their blocked coins will be deleted. As a result, *(n - 2) rn* will achieve a consensus.

We continue this process till $state_{n-1}$ in which:

$state_{n-1}$: (n - 1) of them do not participate in voting process within 24 hours, then (n - 1) *rn* will be removed from RNset and their physical address will be blocked forever and their blocked coins will be deleted. As a result, one *rn* determines the result.

And finally $state_n$:

$state_n$: All of registered nodes do not participate in voting process within 24 hours and all of them will be removed from RNset and after joining new registered nodes we will have another new set of registered nodes, then we go to $state_1$. Thus eventually we achieve a consensus.

Note that since adversary's resources is limited (i.e. coins and physical addresses) thus after time of $\Delta t$ that is a limited time, as a result set of registered nodes after a limited time will achieve a consensus.

### 5.3 Preventing Double-Spending Attack by RDV

In RDV, Double-Spending is impossible, because every transaction needs for vote of all registered node. As a result, while majority of registered nodes are honest (i.e. the safety condition in RDV) every coin that is spent more than one time is detected by the priority table 1 and equation 1.

$$if\ tx_i \rightarrow (coin + add) == tx_j \rightarrow (coin + add)\ then \tag{1}$$

$$a\ double\text{-}spending\ occurred$$

where *tx* transaction, *add* is address of sender and '+' is concatenation operation.

### 5.4 Preventing Intentional Fork by RDV

In Bitcoin, the priority parameter for choosing between two chains is the chain with more difficulty. Thus, always the longest chain is selected by the nodes. As a result, forking blockchain, at least temporarily, is possible (see Figure 1). Whereas, in RDV protocol, since every transaction needs for the vote of all register nodes to be confirmed, in case of removing a transaction from the blockchain, the process is the same. This means that for removing a block including related transaction from the blockchain needs for a voting process in which all registered nodes participate. As a result, while the safety threshold is met (i.e. majority of registered nodes are honest) forking blockchain is impossible.

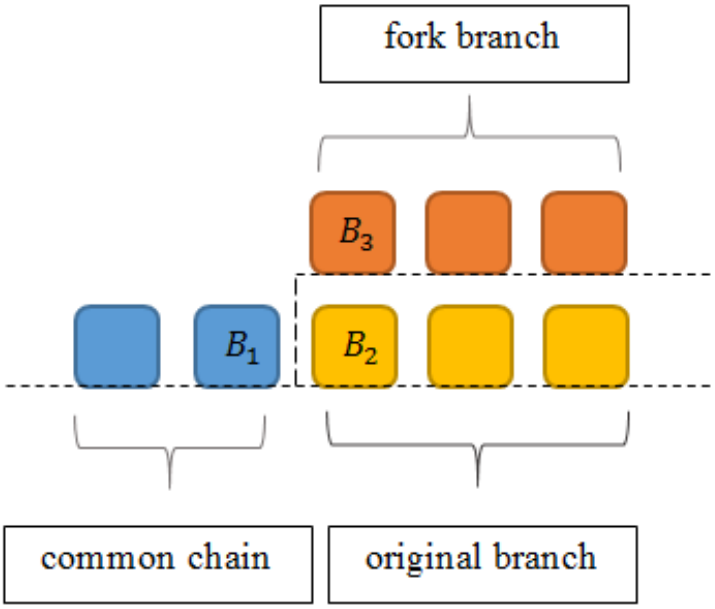


**Fig. 1.** Forking blockchain.

### 5.5 Preventing Accidental Fork by RDV

We removed the Poisson nature of proof-of-work and in Priority Point table is determined that next block is dedicated to a particular transaction (e.g. $tx_i$). Thus, it is not possible two transactions are committed at the same time (see Figure 2). Note that in the case two transactions have same priority point, then the one which has been transfered first, it has more priority point to participate in voting process (see table 1).

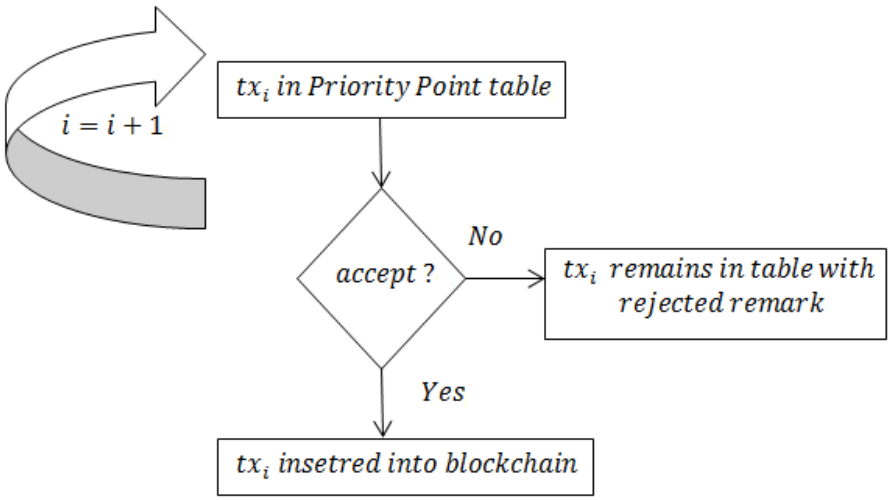


**Fig. 2.** Transaction participation in voting process.

### 5.6 Removing Parameters For Being Winner: Fastest, Longest, Most Difficult

In Bitcoin system, miners try to adjust their strategy in a speed game in which they must produce a chain with most difficulty that is the longest chain. All these parameters lead to a motivation for rational miners to design a block-withholding attack and forking blockchain, since always longest chain that has more difficulty is accepted as a part of blockchain. On the other hand, according to Nash equilibrium in which miners choose a strategy by which they must deliver their chain entire the network, this leads to choose less transactions with greatest fee. Since transactions with greater amount have greatest fee, so they will be chosen by miners and as a result micro-payments must wait for a long term to be inserted in blockchain.

*Transactions Priority for Voting:* For supporting better micro-payments in the network, in RDV nodes do not decide which transaction must be participated in voting process, but also there is a parameter i.e. ***Priority Point*** which calculated as follow:

$$Priority\ Point = TxAmount + (CurrentTime - TxTimeStamp) \quad (2)$$

Where, *TxAmount* is the amount of transaction and *TxTimeStamp* is the time at which transaction has been done. Then transaction with most ***Priority Point*** will be chosen for voting process. In such a situation, for example, a transaction with the amount of 20 coins that has been waited 5 minutes has the same ***Priority Point*** of a transaction with the amount of 5 coins that has been waited 20 minutes for confirmation.

| $Priority$ | $Tx$ | $Coin$ | $SenderAddress$ | $PriorityPoint$ |
|---|---|---|---|---|
| 1 | $tx_i$ | $coin_i$ | $add_i$ | $max$ |
| 2 | $tx_j$ | $coin_j$ | $add_j$ | ... |
| 3 | $tx_k$ | $coin_k$ | $add_k$ | ... |
| ... | ... | ... | ... | ... |
| $n$ | $tx_m$ | $coin_m$ | $add_m$ | $min$ |

**Table 1.** Transactions Priority for Voting.

### 5.7 Discussion on RDV Algorithm:

What happens when a user attempts to cheat and presents an old timestamp to increase the Priority Point of his transaction? Since the information propagation is calculable (e.g. $m$ minute) so, if a rational node intends to forge the timestamp to e.g.$m$ + *10* minutes ago, then the question of honest nodes is why this transaction has not been broadcast 10 minutes ago (i.e. immediately after doing transaction) ? Thus, there is some issues in this transaction Moreover rational node is able to forge TS, if the other side (receiver or sender) is rational as well.

How to bootstrap such a system? the initial deposit is negative deposit. It means that initially a voter deposits -$d$ coins. Then, in case of winning he gets rewards and so he has enough coins for the next time and if he has to pay some penalty, the first time he receives some coins (e.g. $r$ coins), then he will have $r$ - $d$ coins.

In order to hack the network and gain control over the system, more than 50% of the registered computers must be controlled at the same time. This would require hacking thousands of computers at once, which is almost impossible given the size of the network.

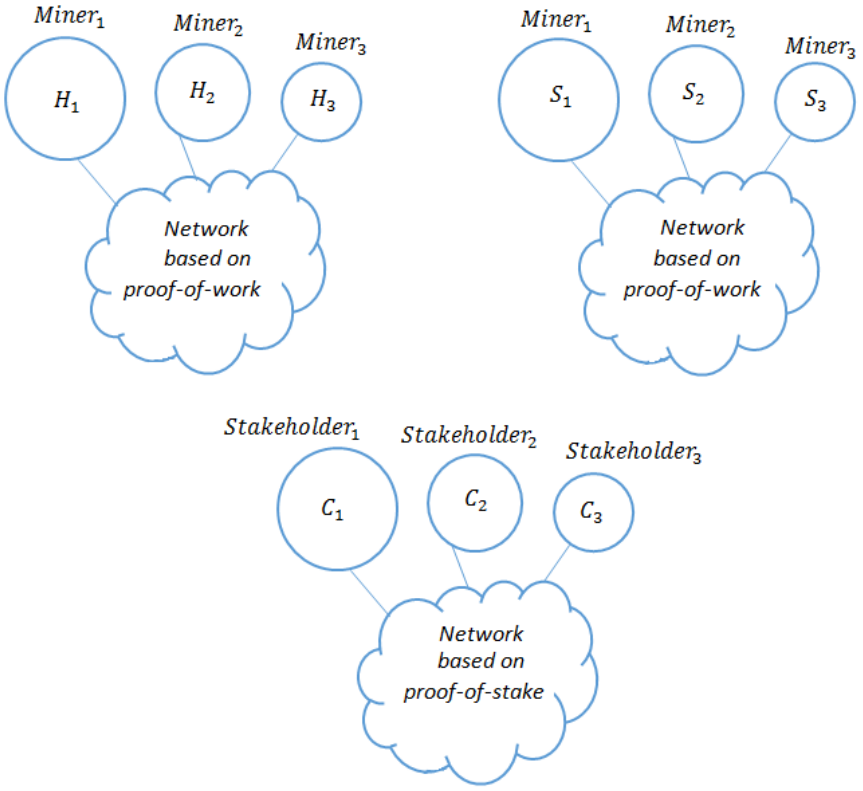


**Fig. 3.** H: hashing power , S: storage space , C: percentage / amount of coins. In PoW: (1) $H_1 > H_2 > H_3 \implies Miner_1 controlpower > Miner_2 controlpower > Miner_3 controlpower$ and (2) $S_1 > S_2 > S_3 \implies Miner_1 controlpower > Miner_2 controlpower > Miner_3 controlpower$, in PoS: (3) $C_1 > C_2 > C_3 \implies Stakeholder_1 controlpower > Stakeholder_2 controlpower > Stakeholder_3 controlpower$. (1)(2)(3) $\implies$ *network tends towards centralization* $\implies$ *decentralization is collapsed.*

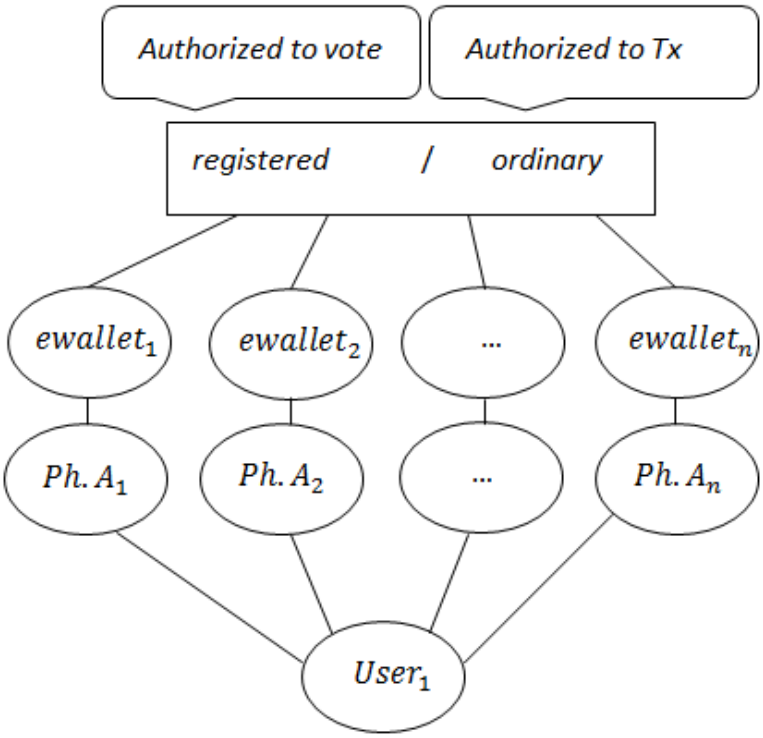


**Fig. 4.** A user architecture in RDV model.

## References


1. Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." Consulted 1.2012 (2008): 28
2. Back, Adam. "Hashcash-a denial of service counter-measure." (2002)
3. Decker, Christian, and Roger Wattenhofer. "Information propagation in the bitcoin network." Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on. IEEE, 2013
4. Eyal, Ittay, and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable." Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2014. 436-454
5. Garay, Juan, Aggelos Kiayias, and Nikos Leonardos. "The bitcoin backbone protocol: Analysis and applications." Advances in Cryptology-EUROCRYPT 2015. Springer Berlin Heidelberg, 2015. 281-310
6. Kroll, Joshua A., Ian C. Davey, and Edward W. Felten. "The economics of Bitcoin mining, or Bitcoin in the presence of adversaries." Proceedings of WEIS. Vol. 2013. 2013
7. Bonneau, Joseph, et al. "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies." 2015 IEEE Symposium on Security and Privacy. IEEE, 2015
8. J. Aspnes, C. Jackson, and A. Krishnamurthy. Exposing computationally-challenged Byzantine impostors. Technical report, Yale, 2005
9. M. Okun. Agreement among unacquainted Byzantine generals. In Distributed Computing. 2005
10. https://en.bitcoin.it/wiki/WeaknessesSybilattack
11. Babaioff, Moshe, et al. "On bitcoin and red balloons." Proceedings of the 13th ACM conference on electronic commerce. ACM, 2012
12. Ben Laurie. Decentralised Currencies Are Probably Impossible
13. Lerner, S. D. "Even faster block-chains with the DECOR protocol." (2014)
14. Sompolinsky, Yonatan, and Aviv Zohar. "Accelerating bitcoin's transaction processing fast money grows on trees." Not Chains (2013)
15. Miller, Andrew, and Joseph J. LaViola Jr. "Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin." Available on line: http://nakamotoinstitute. org/research/anonymous-byzantine-consensus (2014)
16. Bamert, Tobias, et al. "Have a snack, pay with Bitcoins." Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on. IEEE, 2013
17. Karame, Ghassan O., Elli Androulaki, and Srdjan Capkun. "Double-spending fast payments in bitcoin." Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012
18. King, Sunny, and Scott Nadal. "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake." self-published paper, August 19 (2012)
19. Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum Project Yellow Paper 151 (2014)
20. Mazieres, David. "The stellar consensus protocol: A federated model for internet-level consensus." Stellar Development Foundation (2015)
21. McConaghy, Trent, et al. "BigchainDB: A Scalable Blockchain Database." (2016)
22. Schwartz, David, Noah Youngs, and Arthur Britto. "The Ripple protocol consensus algorithm." Ripple Labs Inc White Paper 5 (2014)
23. Dwork, Cynthia, Nancy Lynch, and Larry Stockmeyer. "Consensus in the presence of partial synchrony." Journal of the ACM (JACM) 35.

24. Kwon, Jae. "Tendermint: Consensus without mining." URL http://tendermint.com/docs/tendermint-v04. pdf (2014)
25. Eyal, Ittay. "The miner's dilemma." Security and Privacy (SP), 2015 IEEE Symposium on. IEEE, 2015.
26. Solat, Siamak, and Maria Potop-Butucaru. "ZeroBlock: Preventing Selfish Mining in Bitcoin." arXiv preprint arXiv:1605.02435 (2016)
27. Solat, Siamak, and Maria Potop-Butucaru. "ZeroBlock: Timestamp-Free Prevention of Block-Withholding Attack in Bitcoin." arXiv preprint arXiv:1605.02435 (2016)
28. Solat, Siamak. "Security of Electronic Payment Systems: A Comprehensive Survey." arXiv preprint arXiv:1701.04556 (2017)
29. https://litecoin.org/