TomoChain: Masternodes Design

Technical White Paper Version 1.0

May 30, 2018

TomoChain R&D Team

TomoChain Pte. Ltd.

Email: contact@TomoChain.com[*]

Abstract

In this paper, we present an overview architectural design of TomoChain technology and solutions. TomoChain is a public EVM (Ethereum Virtual Machine)-compatible blockchain with the following advantages: low transaction fee, fast confirmation time, double validation and randomization for security guarantees. In particular, we propose Proof-of-Stake Voting (PoSV) consensus, a Proof-of-Stake (PoS)-based blockchain protocol with a fair voting mechanism, rigorous security guarantees and fast finality. We also present a novel reward mechanism and show that, with this mechanism, the blockchain has a low probability of forks, fast confirmation times, plus the contributions and benefits of masternodes are fair in the sense that the probability distribution function is uniform eventually.

Index Terms

Blockchain, Ethereum, TomoChain, Proof-of-Stake Voting, Masternode, Randomization, Security Protocol.

I. INTRODUCTION

The blockchain industry and the infrastructure of the Internet of Value are being built rapidly around the globe, and to many the atmosphere is eerily similar to the building of the Internet in the late '90s, with pioneers and dreamers coming together to build a new future. TomoChain can be a leading part of this phenomenon through seamlessly merging an ecosystem of applications with cryptographic tokens used by millions of mainstream users with a unique blockchain infrastructure architecture allowing for a fast, secure, frictionless payment and trusted store of value.

Distributed systems have been researched in a "*permissioned setting*" where the number of participants in the system and their identities are common knowledge. In 2008, Satoshi Nakamoto - "proposed his celebrated "blockchain protocol" which attempts to achieve consensus in a permissionless setting: anyone

---

[*]Comments and feedbacks are highly appreciated, but all errors and mistakes belong to the authors.

can join (or leave) the protocol execution (without getting permission from a centralized or distributed authority), and the protocol instructions do not depend on the identities of the players" [10]. Later on, Ethereum with its Ethereum Virtual Machine (EVM) proposed several significant enhancements compared to Bitcoin, including Smart Contracts. Both Bitcoin and Ethereum have some issues, especially with transaction processing performance. In order to construct an efficient and secured consensus protocol for TomoChain, we tackle the following main bottlenecks of classic blockchains:

- **Efficiency:** Existing blockchains as employed by major crypto-currencies(e.g., Bitcoin or Ethereum) do not scale well to handle a large transaction volume, e.g. Bitcoin and Ethereum can handle around 10 transactions/second. This small throughput severely hinders a wide-spread adoption of such crypto-currencies.

- **Confirmation times:** The 10 minutes Bitcoin block-time [1] is significantly larger than network latency. Furthermore, a Bitcoin block requires 5 subsequent blocks following it so that it can be confirmed; thus it takes on average one hour for a transaction to be confirmed (with low confidence). While Ethereum uses a smaller block-time, the average confirmation time still remains relatively high, around 13 minutes [6], [9]. These long confirmation times hinder many important applications (especially smart contract applications).

- **Fork Generation:** The problem of fork chain consumes computational energy, time, and creates potential vulnerabilities for different types of attacks.

With the motivation as mentioned above, our persistent and ultimate goal of research is to propose the consensus protocol focusing on the following key strategies:

- Double Validation to strengthen security and reduce fork
- Randomization to guarantee the fair and prevent handshaking attack
- Fast confirmation time and efficient checkpoints for finality or rebase

To start dealing with these problems, in this paper, we present an overview architectural design of TomoChain's master nodes. In particular, we propose Proof-of-Stake Voting (PoSV) consensus, a Proof-of-Stake (PoS)-based blockchain protocol with a fair voting mechanism, rigorous security guarantees and fast finality. We also present a novel reward mechanism and show that, with this mechanism, the blockchain has a low probability of forks, fast confirmation times, plus the contributions and benefits of masternodes are fair in the sense that the probability distribution function is uniform eventually.

Structure of the remainder of the paper. Section II-A explains the intuition ideas and overview architectural design of masternodes, framework and background protocols that help mass readers (e.g., investors, traders, others) who may not have technical knowledge understand our mechanism easily.

Section II-B presents TomoChain stakeholder policy, masternode committee voting systems, and reward mechanism. Section II-C explains the motivation and double validation process as well as finality checkpoint of the protocol. In Section II-D, we present the formalization of our model in a mathematical way to show the soundness of our model and protocol. Section III discusses the security analysis and resistant strain of potential attacks. We discuss and compare TomoChain with several existing blockchains in Section IV. Finally, we conclude the paper in Section V.

II. TOMOCHAIN MASTERNODE DESIGN

A. The TomoChain architecture

The TomoChain blockchain is produced and maintained by a set of *masternodes* in a consistent manner through the TomoChain consensus protocol as shown in Fig. 1. These masternodes are full nodes that hold TOMO . For a coin-holder to become a masternode, two requirements must be satisfied:

- The coin-holder must hold at least a minimum required amount of coin (see next section for more details).
- The coin-holder must be one of the most voted masternode candidates in the system. The voting by coin-holders is credited through a *Voting DApp* that allows coin-holders to *send TOMO through the smart contract mechanism*.

In addition to the voting system which is an improvement over the current Bitcoin and Ethereum blockchain, TomoChain also provides a new technique, namely *Double Validation* complemented with a *Randomization* mechanism. This new technique significantly decreases the probability of having invalid blocks in the blockchain. These enhancements and the components of TomoChain are step-by-step detailed in the followings.

B. Stakeholders & Voting

Coin Holders, Masternodes

Coin-holder is as simple as its name: users who join the network, who own and transfer TOMO . Masternodes are full-nodes which maintain a copy of the blockchain, produce blocks and keep the chain consistent. It is worth noting that, TomoChain does not have miners as in the current Proof-of-Work-based blockchain systems such as Bitcoin and Ethereum. Only masternodes can produce and validate blocks.

Masternodes are selected via a voting system. The first requirement of being masternodes is to deposit 50 000 TOMO to the Voting Smart Contract. Then, these depositors are listed as masternode candidates in the Voting DApp, which allows coin-holders to vote for them by sending TOMO to the smart contract.
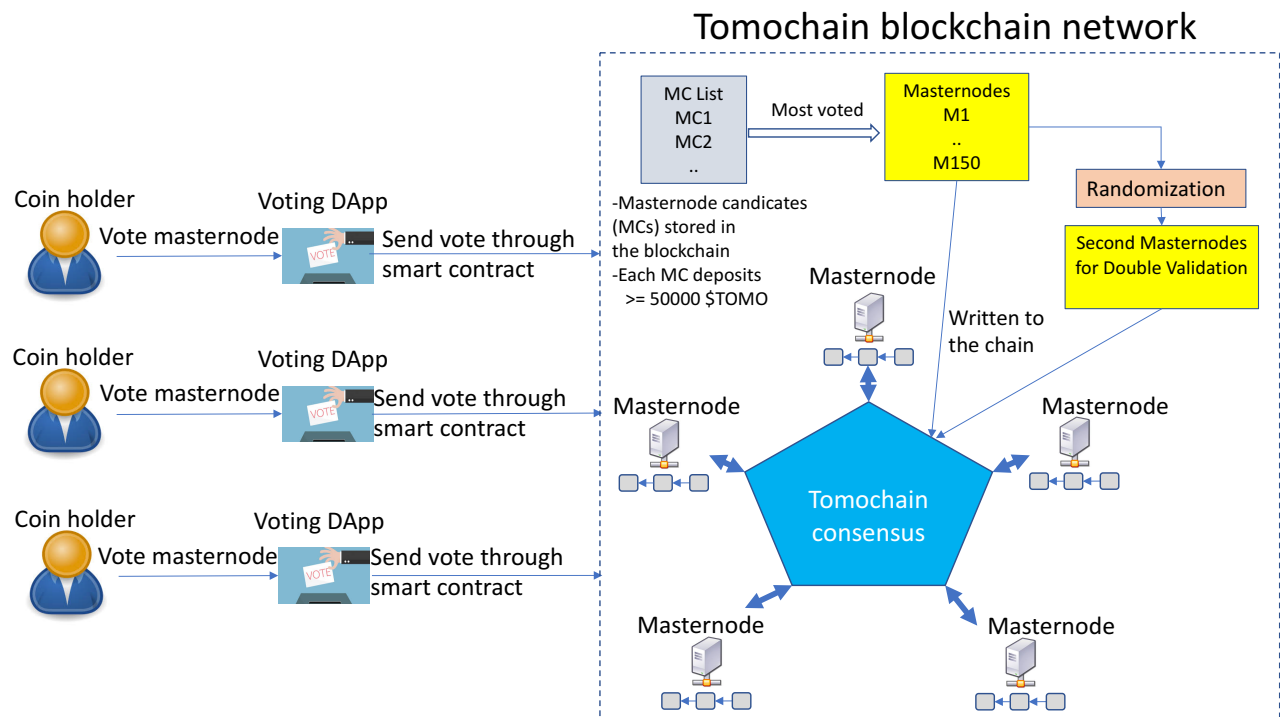
Fig. 1. TomoChain architecture

Masternodes which work hard in the system to create and verify blocks will be incentivized with TOMO . Furthermore, coin holders who vote for these incentivized masternodes will also receive TOMO in proportion to the amount of TOMO they have invested via ballots. TomoChain engineers take responsibility to design that fair, explicit, automated and accountable reward mechanism.

The list of masternode candidates is dynamically sorted based on voted coins. The performance of the masternodes will be tracked and reported back to the coin holders in terms of three main metrics: CPU/Memory charts which ensure the workload of the masternodes, the number of signed blocks which indicates their work performance and the last signed block which figures out their last activity. Coin-holders, at any time, can unvote masternodes, who have low performance, and give their votes to the other masternodes who have beter performance. Coin-holders have incentives to do that because their voted coins are seen as investment to their supported masternodes, thus they should choose a voting strategy in order to maximize their profit from the investment.

This simple trick keeps the system healthy since masternodes always have to race for their position so that all weak masternodes will eventually be eliminated. Therefore, only the strongest masternodes are voted and can flourish.

Voting & Masternode Committee

There are maximum ninety-nine masternodes elected in the masternode committee. The required amount of deposit for masternode role is set at 50 000 TOMO . This amount is locked in a *voting smart contract*. Once a masternode is demoted (by not remaining in the top ninety-nine voted masternodes) or intentionally quits the masternode candidates list/masternode committee, the deposit will have been locked for a month.

Coin-holders can vote at any time, by any number of votes (which is actually counted by the amount of TOMO they bet on some masternode candidates). They can use masternode's performance statistics in the governance *Voting DApp* as reference information to give votes. The set of masternodes is dynamically sorted by the amount of TOMO and counted up to ninety-nine, upon reception of votes.

Reward Mechanism

For each iteration of 900 blocks (called epoch), a checkpoint block is created, which implements only reward works. The masternode, who takes turn in the circular and sequential order to create block, has to scan all of the created blocks in the epoch and count number of signatures. The reward mechanism is designed following the policy as follow: the higher number of signatures one masternode has made, the more reward he earns. For instance, within an epoch, masternode A who has sealed twice the blocks than masternode B earns double amount of TOMO than masternode B does.

Furthermore, there is also a reward sharing ratio among coin-holders and masternode who has been elected supported by the coin-holders. Specifically, each epoch consists of 900 blocks, which will reward a total of 250 TOMO in the first two years. This amount of 250 TOMO will be divided to all of the Masternodes proportionally to the number of signatures they sign during the epoch. Afterward, the reward achieved by each Masternode will be divided into three portions.

- **Infrastructure Reward**: The first portion of 40% called **Infrastructure Reward** goes to the Masternode.
- **Staking Reward**: The second portion of 50% called **Staking Reward** goes to the pool of all voters for that Masternode which is shared proportionally based on the token stake.
- **Foundation Reward**: The last portion of 10% called **Foundation Reward** goes to a special account controlled by the Masternode Foundation, which is run by TomoChain company initially.

It is worth noting that coin-holders who unvote before the checkpoint block will not receive any shared reward in the **Staking Reward** portion.

C. TomoChain Consensus Protocol

Double Validation Process

In TomoChain, masternodes share equal responsibility to run the system and keep it stable. Full nodes should run on powerful hardware configuration and high-speed network connectivity in order to ensure the required block time (target to two seconds). Only masternodes can produce and seal blocks. In order for that, the TomoChain consensus relies on the concept of **Double Validation** that improves some existing consensus mechanisms, namely **Single Validation**. In the followings, we first describe the **Double Validation**, then analyze the differences and improvements of **Double Validation** compared to **Single Validation**.

Double Validation (DV): Similar to some existing PoS-based blockchains such as Cardano, each block is created by a block producer, namely masternode, that takes its block creation permission turn following a pre-determined and circular sequence of masternodes for each epoch. However, differently from these existing blockchains, DV in TomoChain requires the signatures of two masternodes on a block to be able to push the block to the blockchain. One of the masternodes is the **block creator** while the other one, namely **block verifier** is randomly selected among the set of voted masternodes that validates the block and signs it. In the followings, for more convenience, **block creator** and **block verifier** are used interchangeably for the masternode 1 (block producer) and the randomly selected masternode 2 for a block, respectively. The process of randomly selecting the block verifiers is detailed in the next paragraphs. Note that, there is no mining in the block creation as in Proof-of-Work-based blockchains (e.g. Ethereum and Bitcoin). It means that a created block is valid if and only if it is sealed by enough two signatures from a block creator and a corresponding block verifier to confirm the correctness of it.

We believe this DV technique enhances the stability of the blockchain by diminishing the probability of producing "garbage" blocks while still maintaining the system security and consistency. Randomization of block verifiers in DV is the key factor of reducing risks coming from paired masternodes trying to commit malicious blocks. Furthermore, comparing to some current public blockchains in the market, by utilizing the DV technique, TomoChain brings significant improvements in the block time by only requiring two signatures per block. For the purpose of showing our enhancement over existing PoS-based blockchains, we analyze the differences between DV and the Single Validation mechanism in some existing blockchains as follows.

Improvements of Double Validation over Single Validation: Let's show the improvements of DV compared to Single Validation through analyzing some attacking scenarios as shown in Fig. 2 and Fig. 3.
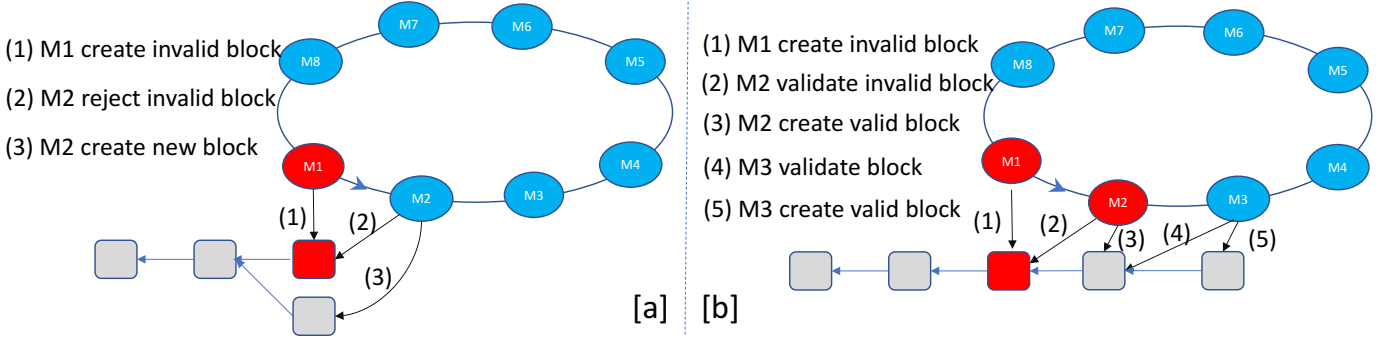
Fig. 2. Single Validation (SV): (a) SV with block creation masternode as an attacker and (b) SV with two consecutive block creation masternodes as attackers

- **Single Validation** In Single Validation, in an epoch, each masternode, e.g. M1, sequentially takes its turn to create a block, e.g. block100. The next masternode, e.g. M2, in the sequence then validates the created block100. If block100 is invalid (that potentially means that M1 is an attacker) and contains a transaction that invalidly benefits M1, if M2 is honest (see Fig. 2 [a]), it rejects block100 and creates another block100 next to block99. But, if M2 is an attacker (see Fig. 2 [b]) that corporates with M1, M2 ignores the invalidation of block100, signs it and creates next block, namely block101 that is valid. Then, the next masternode M3 verifies that block101 is valid, M3 signs block101 and creates a block102. By this way, Single Validation potentially leaves the blockchain with "garbage" or invalid blocks which require a "rebase" to restore the validity of the blockchain.

- **Double Validation** We claim that our DV technique significantly reduces the probability of having garbage blocks in the blockchain. Assuming that M1 and M2 are the block creator and block verifier, respectively, for block100 in our DV. If block100 is invalid and M2 is honest (see Fig. 3 [a]), M2 will not seal this block. Therefore, the next block creator M3 for creating block101 will see that block100 does not have enough 2 signatures, thus reject block100 and create another block100 next to block99. On the other hand, if M2 is also an attacker pairing/handshaking with M1 (see Fig. 3 [b]), M2 signs block100 despite its invalidity (remember that the block verifier M2 is randomly selected, there has little chance of successfully pairing M1 and M2). Next, even though M3 will verify that block100 has two valid signatures, M3 still rejects it because block100 is invalidated by M3 that will create another valid block100. In order to break the stability and consistency of the blockchain in this case, M3 should be an attacker together with M1 and M2, which, however, has a very low probability. In other words, DV strengthens the consistency of the blockchain and makes it hard to break.
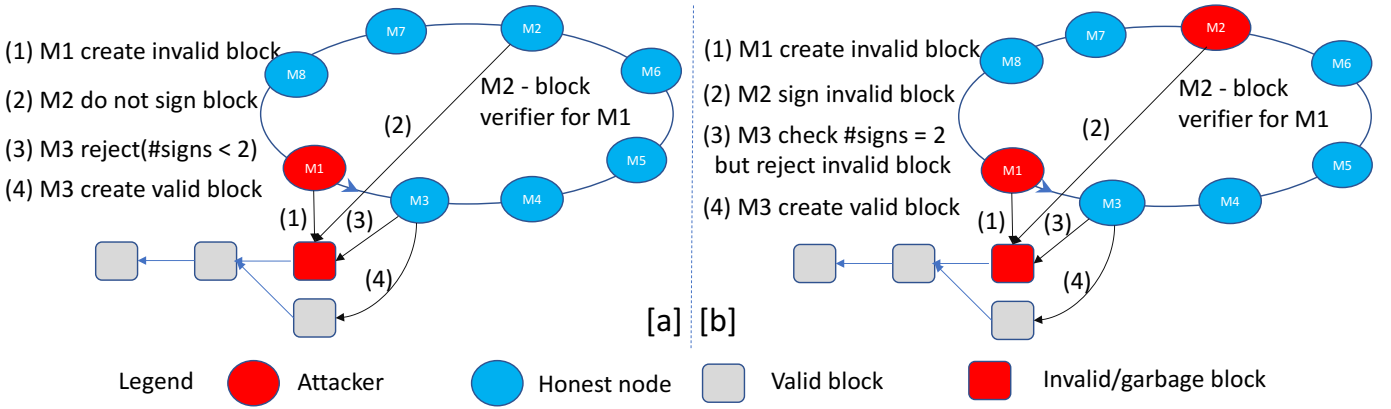
7

Fig. 3. Double Validation (DV): (a) DV with block creator as an attacker and (b) DV with both block creator and block verifier as attackers

Randomization for Block Verifiers for Double Validation

The First Masternode/Block Creator: The first masternode/block creator in a given epoch e can be selected by a round-turn game and can be formal defined as an array:

$$
\begin{bmatrix} \nu_1 \end{bmatrix} = \begin{bmatrix} V_{1.1}^e \\ V_{1.2}^e \\ . \\ . \\ . \\ V_{1.n-1}^e \\ V_{1.n}^e \end{bmatrix}
$$

Random Matrix and Smart Contract: Let m be the number of masternodes, n be the number of slots in an epoch. In order to randomly generate the block verifiers for the next epoch $e+1$, the process is performed by the following steps.

- **Step 1: Random Numbers Generation and Commitment Phase:**

  First, at the beginning of epoch e, each masternode V_i will securely create an array of $n+1$ special random numbers $Recommend_i = [r_{i.1}, r_{i.2}, ..., r_{i.n}, \theta_i]$, where $r_{i.k} \in [1, ..., m]$ indicating the recommendation of ordered list of block verifiers for the next epoch of V_i, and $\theta_i \in \{-1, 0, 1\}$ is used for increasing the unpredictability of the random numbers. Second, each masternode V_i has to encrypt the array $Recommend_i$ using a secret key SK_i, say $Secret_i = Encrypt(Recommend_i, SK_i)$ as the encrypted array. Next, each masternode forms a "*lock*" message that contains encrypted array $Secret_i$; signs off this message with its blockchain's private key through the Elliptic Curve Digital Signature

Algorithm (ECDSA) scheme currently used in Ethereum and Bitcoin along with the corresponding epoch index and its public key generated from its private key. By doing this, every masternode can check who created this *lock* message through ECDSA verification scheme and which epoch it relates to. Then, each node V_i sends their *lock* message with its signature and public key to a **Smart contract** stored in the blockchain, so that eventually each masternode collects and knows the *lock*s from all other masternodes.

- **Step 2: Recovery Phase:** The recovery phase is for every node to reveal its previous *lock* message so that other nodes can get to know the secret array it has sent before. A masternode only starts revealing its lock message if all masternodes have sent their lock messages to the smart contract or a certain timeout event occurs. Each masternode then opens its lock message by sending an "*unlock*" message to the smart contract for other masternodes to open the corresponding *lock*. Imagine a *commitment*-like scheme in this case where a *lock* message is a commitment message locking its contained recommendation array $Recommend_i$ (so that no one can open or guess the contained array), and the *unlock* message gives the key for other masternodes to decrypt the box and retrieve the values of $Recommend_i$. Eventually, a masternode has both *lock*s and *unlock*s of others. If some elector is an adversary which might publish its *lock* but not intend to send the corresponding *unlock*, other masternodes can ignore the adversary's *lock* and set all its random values be 1 as default. The idea is simple: the network can keep working successfully even if some masternodes are adversaries.

- **Step 3: Assembled Matrix and Computation Phase:** At the point of the slot n^{th} of the epoch e, the secret arrays $Secret_i$ in the smart contract will be decrypted by each masternode and return the plain version of $Recommend_i$. Each tuple of the first n numbers of each V_i will be assembled as the ith column of an $n \times m$ matrix. All the last number θ_i forms a $m \times 1$ matrix. Then each nodes will compute the block verifiers ordered list by some mathematical operations as explained below. The resulting output is a matrix $n \times 1$ indicating the order of block verifiers for the next epoch $e + 1$.

The Second Masternode/Block Verifier: Then, each node soon compute the common array ν_2 for the order of the block verifiers by the following steps as in Equation 1.

$$
\begin{bmatrix} \nu_2' \end{bmatrix} = \begin{bmatrix} v_{2.1}^{e+1} \\ v_{2.2}^{e+1} \\ \vdots \\ v_{2.n}^{e+1} \end{bmatrix} = \begin{bmatrix} r_{1.1} & r_{2.1} & \cdots & & r_{m.1} \\ r_{1.2} & r_{2.2} & \ddots & & \vdots \\ r_{1.3} & \ddots & \ddots & & r_{m.3} \\ \vdots & & & r_{m-1.n-1} & r_{m.n-1} \\ r_{1.n} & \cdots & & r_{m-1.n} & r_{m.n} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_m \end{bmatrix} \tag{1}
$$

$$\begin{bmatrix} \nu_2 \end{bmatrix} = \begin{bmatrix} \nu_2' & mod & m \end{bmatrix} = \begin{bmatrix} \left| v_{2.1}^{e+1} \right| & mod & m \\ \left| v_{2.2}^{e+1} \right| & mod & m \\ \vdots \\ \left| v_{2.n}^{e+1} \right| & mod & m \end{bmatrix} \qquad (2)$$

Then, ν_2 is obtained by modulo operation of element values of ν_2' as in Equation 2:

Finality Analysis

"There is a standard definition of "total economic finality": it takes place when $\frac{3}{4}$ of all masternodes make maximum-odds bets that a given block or state will be finalized. This condition offers very strong incentives for masternodes to never try colluding revert the block: once masternodes make such maximum-odds bets, in any blockchain where that block or state is not present, the masternodes lose their entire deposit." [13]

TomoChain keeps that standardization in the design so that one block is considered as irreversible if it collects up to $\frac{3}{4}$ signatures of all masternodes committee. The time-line of blockchain creation process, checking finality and mark the block as immutable is described as in Figure 4 below.
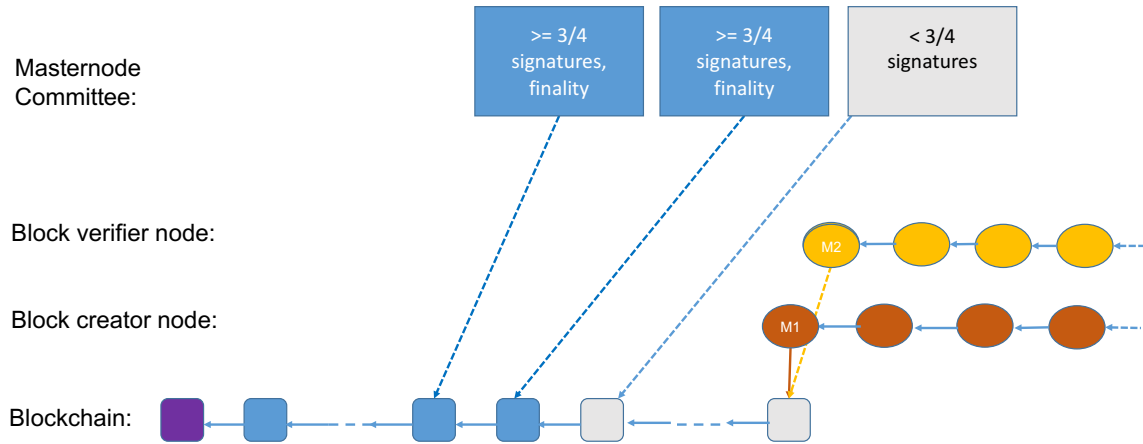

Fig. 4. Timeline of Blockchain Making Process

D. Consensus Protocol: <mark>Formalization</mark>

Basic Concepts & Protocol Description

In order to have a solid foundation for us to prove that our blockchain can achieve what are claimed, we first present our preliminary formalizations of the concepts that will be used in our yellow paper later.

To start, as we are dealing with proof of stake consensus algorithm, we follow the way of formalization in the recent works in the literature like Cardano [6] and Thunder Token [8], [10]. In particular, we recall the following concepts and definitions that were presented in [6] and adapt them to the context of TomoChain.

Time, Slots, Epoch

As previously described, ideally, each epoch is divided into 900 block time, that is called block *slot*. Only one block can be created in a slot. We assume that there is a roughly synchronized clock that allows for masternodes to learn the current slot. This simplification will effectively permit masternodes to execute the signing and validation process of the PoSV consensus, where each masternode must collectively create a block to the current slot. For more simplification, each slot sl_r is accessed by an integer $r \in \{1, 2, ...\}$, and suppose that the real time window that corresponds to each slot has the following properties, which are similar to what are specified in Cardano.

1) Every masternode can determine the index of the current slot based on the current time and "any discrepancies between parties' local time are insignificant in comparison with the length of time represented by a slot" [6].

2) The amount of a slot time is sufficient to guarantee that any message transmitted by an honest party at the beginning of the time window will be received by any other honest party by the end of that time window. While this assumption similar to Cardano, TomoChain requires it in order for a block creator to propagate its created block to the corresponding block verifier to ensure that the block is signed by both the masternodes before the next block creator builds another block on top of it.

As mentioned in Section II-A, in our setting, we assume that the fixed set of m (150) masternodes $V_1, V_2, ..., V_m$ interact throughout the protocol to reach the consensus. For each V_i a public/private key pair (pk_i, sk_i) for a prescribed signature scheme, ideally ECDSA, is generated. Furthermore, we assume that the public keys $pk_1, .., pk_m$ of the masternodes are distributed and known by all of them (that means a masternode knows all public keys of other nodes). Some notable definitions of the blockchain concepts are defined following the notation of [11].

Definition 1 (State): A state [6] is an encoded string $st \in \{0, 1\}^{\lambda}$.

11

Definition 2 (Block): A block [6] B generated at a slot sl_i contains the current state $st \in \{0,1\}^\lambda$, data $d \in \{0,1\}^*$, the slot number i and a signature $\Sigma = Sign_{ski}(st, d, sl_i)$ computed under sk_i corresponding to the masternode V_i generating the block.

Algorithm 1: Algorithm illustrated the consensus protocol

Input: m - Number of masternodes, n number of slots in an epoch
Output: The ledger of the blockchain C
begin
    Create the empty blockchain (stack) C;
    Initiate ICO; coinholders;
    Voting for the masternode committee (master nodes) $VC \leftarrow \{V_1; V_2; ..., V_m\}$;
    Initiate the first epoch $e_1 \leftarrow \{sl_1, sl_2, ..., sl_n\}$;
    Randomly generate the array of second masternodes for the first epoch
     $SV_1 \leftarrow [v_{2.1}^1, v_{2.2}^1, ..., v_{2.n}^1]$;
    Create the genesis block B_0;
    Update the blockchain $C \leftarrow C.push(B_0)$;
    while true **do**
        while j is less than n **do**
            Create block B_j by the first masternode;
            Update the blockchain $C \leftarrow C.push(B_j)$;
            Validate the block B_j by the second masternode;
            Broadcast and validate the block B_j by VC_i;
            if B_j has more than 3/4 masternode committee members sign **then**
                FINALITY$(B_j.ID)$ = true;
            if j = n **then**
                $j \leftarrow 1$;
            else
                j++;
        if len(C) mod n = 0 **then**
            doCheckpoint();
            Voting for the masternode committee for the next epoch $VC \leftarrow \{V_1; V_2; ..., V_m\}$;
            Random generate the array of verifier masternodes for the next epoch $(i+1)^{th}$;
            $SV_{i+1} \leftarrow [v_{2.1}^{i+1}, v_{2.2}^{i+1}, ..., v_{2.n}^{i+1}]$;
            $e_{i+1} \leftarrow i * n * 2 + e_1$;
            i++;

Definition 3 (Blockchain): A blockchain [6] C is a sequence of blocks $B_1, ..., B_n$ associated with a strictly increasing sequence of slots for which the state st_i of B_i is equal to $H(B_{i-1})$, where H is a collision-resistant cryptography hash function. A blockchain has a number of properties, including the length of a chain $len(C) = n$, which is its number of blocks, and the block B_n is the head of the chain, denoted $head(C)$.

As mentioned earlier, in our TomoChain model, we set each time *slot* sl_i as 2 seconds; an epoch is a set R of 900 slots $\{sl_1, sl_2, ..., sl_{900}\}$ (an epoch time duration equals to 1800 seconds).

In summary, the consensus protocol of TomoChain can be formalized in Algorithm 1. The Algorithm 1 is simulated and explained as a process shown in Fig. 5.
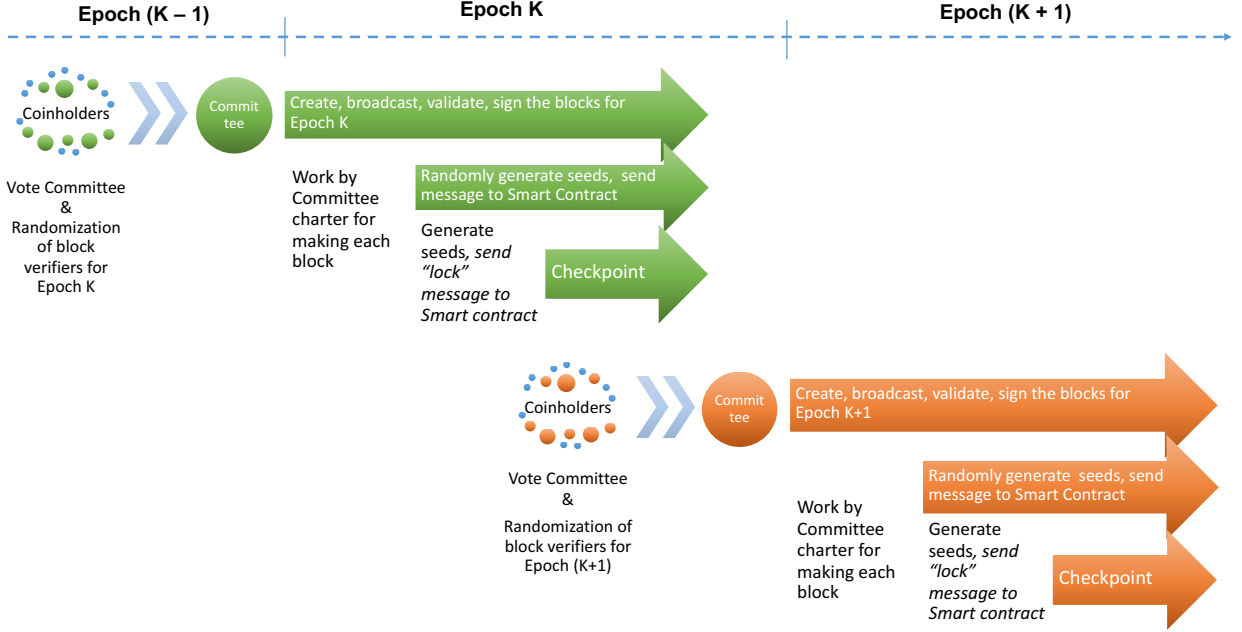

Fig. 5. Process of Voting Committee, Randomization of Block Verifiers, Creating and Validating Blocks in Each Epoch

III. SECURITY ANALYSIS

Nothing-at-stake

Nothing-at-stake is a well-known problem in PoS-based blockchain, just like 51% attack in PoW algorithm. PoW-based miners require CapEx (capital expenditures) for buying mining equipment such as ASICs and OpEx (operation expenditures) such as electricity to solve mathematical puzzles securing the network [18]. That means, there is always an intrinsic cost for miners in mining regardless of its success. In case of a fork, miners therefore always allocate their resource (equipment) to the chain that they believe is correct in order to get incentives for compensating the intrinsic costs in mining.

In the contrary, in PoS-based systems without mining, during an ideal execution, for creating a fork only costs, masternodes actually do not incur intrinsic costs, other than roughly some block validation and signing cost. As a result, there's an inherent problem of the masternode having no downside to staking both forks. Therefore, there are actually two issues in the original design of PoS. On one hand, for any

masternode, the optimal strategy is to validate every chain/fork, so that the masternode gets their rewards no matter which fork wins. On the other hand, for attackers/malicious masternodes, they can easily create a fork for double spending.

Let's look back how TomoChain handles these two problems. As a reminder, TomoChain maintains a certain order of masternodes in creating and sealing blocks, in each epoch. For the first issue, random/arbitrary forks are hardly happened because the order of block creation masternodes is pre-determined for each epoch. Furthermore, the Double Validation mechanism eliminates the second issue because even one malicious masternode creates two blocks at his turn, only one block then can be validated by the second randomly selected masternode.

Long-range attack

In TomoChain, block is valid only if it collects double validation and finalized once $\frac{3}{4}$ of masternodes verify. Therefore, as long as the number of attackers or malicious nodes and/or fail-stop nodes is less equal than $\frac{1}{4}$ the number of masternodes, the number of masternodes signing a block is at least $\frac{3}{4}$ the total number of masternodes, which makes the block finalized. Thus, there is no chance for one malicious masternode to create longer valid chain because other masternodes will refuse it.

Censorship Attack

If there are more than $\frac{3}{4}$ malicious masternodes in TomoChain, censorship attack might happen. For example, these masternodes refuse valid blocks or simply become inactive. In this case, chain is stuck.

In fact, masternodes are paid for their effort of correctly working so that the chain is actively updated in a consistent manner. More importantly, becoming masternode means a certain amount of coins is locked, 50 000 TOMO in particular. As a result, in order to control more than $\frac{3}{4}$ masternodes, attackers must hold a considerable amount of TOMO and gain huge support from coin-holders. And because of this, the attackers do not have incentives to do any malicious action to harm the chain.

However, in worst case, TomoChain has to do a soft fork in order to reduce number of masternodes to keep the chain running and figure out slasher mechanisms for those malicious masternodes.

Relay Attack

TomoChain supports EIP155 **(https://github.com/ethereum/EIPs/blob/master/EIPS/eip-155.md)**. Transactions in TomoChain are included $CHAIN\_ID$ specified for different public chains. Table I shows recognized $CHAIN\_ID$s.

TABLE I
CHAINS AND CHAIN_ID

CHAIN_ID	Chain(s)
1	Ethereum mainnet
2	Morden (disused), Expanse mainnet
3	Ropsten
4	Rinkeby
30	Rootstock mainnet
31	Rockstock testnet
42	Kovan
61	Ethereum Classic mainnet
62	Ethereum Classic testnet
1337	Geth private chains (default)
77	Sokol, the public POA Network testnet
99	Core, the public POA Network main network
88	TomoChain Mainnet
89	TomoChain Testnet

Safety and liveness

Safety implies having a single agreed upon chain where there are not two or more competing chains with valid transactions in either [12]. A consensus protocol can be *safe* when blocks have settlement finality, or else probabilistic finality. This last sentence reveals that TomoChain can provide safety because it has a settlement finality.

A consensus protocol is considered *live* if it can eventually propagate and make valid transactions onto the blockchain [12]. An occurrence of a liveness fault is when transaction omission, information withholding, or message reordering, among a number of violations are observed. This type of fault is unlikely to happen in TomoChain because the block creation masternodes list is ordered in a pre-determined way for each epoch, thus if even an attacking masternode omits some transactions, the latter will be processed and validated by the next honest masternode in the next block.

DDOS Attack

Masternodes are encouraged to run in well-known public cloud providers such as AWS, Google Cloud or Microsoft Azure which provides multiple DDOS prevention mechanisms. Even in case that some nodes are attacked or fail-stop, the network still works correctly as long as the number of failing and/or attacked nodes is less than 1/4 of the number of masternodes.

15

Spam Attack

TomoChain keeps the same transaction fee mechanism as Ethereum which is indicated via gasPrice. However, TomoChain supports minimum transaction fee (at 1 wei), which somehow enables spamming that attacker tries to broadcast a huge amount of low fee transactions to the system. However, TomoChain masternodes always sort transactions and pick up only high fee transactions into the proposing block. Thus, spammers have little chance to harm the system.

IV. RELATED WORK

Consensus plays an important role to guarantee the success of distributed and decentralized systems. Bitcoin's core consensus protocol, often referred to as Nakamoto consensus [1], realizes a "replicated state machine" abstraction, where nodes in a permissionless network reach agreement about a set of transactions committed as well as their ordering [16]. However, known permissionless consensus protocols such as Bitcoin's Nakamoto consensus come at a cost. Bitcoin and Ethereum rely on PoW to roughly enforce the idea of "one vote per hashpower" and to defend against Sybil attacks. Unfortunately, PoW-based Bitcoin and Ethereum are known to have terrible performance (Bitcoin's transaction processing performance is at peak of around 7 transactions per second as previously mentioned). Moreover, PoW is much criticized because it costs a lot of electricity energy.

In order to design an efficient and cost-effective consensus protocol in the permissionless model, PoS has been discussed extensively in the Bitcoin and Ethereum forum [2], [3]. A PoS blockchain can substitute the costly PoW in Nakamoto's blockchain while still providing similar guarantees in terms of transaction processing in the presence of a dishonest minority of users, where this "minority" is to be understood here in the context of stake rather than computational power [6]. The Ethereum design Casper [17], published by Buterin & Griffith, provides as its initial version a PoW/PoS hybrid consensus protocol, which might eventually switch to a pure PoS system. As in TomoChain, Ethereum Casper requires that *validators* (term similar to block creators) have to deposit an amount. In fact, some concepts used in TomoChain such as checkpoint blocks are borrowed from Casper.

Our Proof-of-Stake Voting (PoSV) consensus protocol proposed in this paper can be seen as a hybrid model. In particular, first, we apply PoSV with voting and **Double Validation** to create, verify and vote for blocks smoothly and efficiently. Whenever potentials of fork branches are detected, we employ the idea in PoW to select the longest branch with the most votes and discard the other branches. With this hybrid approach, PoSV does not only increase the performance and security of blockchain, but also reduce the fork situation in an efficient and practical manner.

16

Recently, there are several consensus protocol research works that are closely related to TomoChain such as EOS [14] and Ouroboros of Cardano [6]. The mechanism of *voting* for masternodes for reaching consensus is utilized by Bitshares [15] and EOS [14], whose consensus protocol is termed *Delegated Proof-of-Stake* (DPoS). DPoS is similar to the Proof-of-Stake Voting consensus of TomoChain in the sense that masternodes (block creators or *witnesses* in DPoS) are elected through a voting system. However, TomoChain requires that masternodes need to deposit a required minimum amount of TOMO to become a masternode candidate, which puts more pressure on the masternodes to work honestly. Furthermore, the **Double Validation** mechanism of TomoChain lowers the probability of handshaking attacks and having invalid blocks, as previously analyzed. EOS also has a maximum of 21 block producers for each epoch, which is *less decentralized* than TomoChain with a maximum of 150 masternodes elected (and this number of masternodes can be changed based on the decentralized governance through voting).

The research-backed Cardano [6] blockchain solution, namely Ouroboros, with the ADA coin, which is purely based on Proof-of-Stake, promisingly claims to provide rigorous security guarantees. Similarly to TomoChain, Ouroboros has a set of block producers for each epoch for creating blocks and each block producer candidate needs to deposit a minimum amount of stake (an amount of ADA). However, note that, Ouroboros only provides **Single Validation**, while **Double Validation** of TomoChain provides several advantages over Single Validation, as previously analyzed. In Ouroboros, the order of block producers, selected among stakers, is based on a biased randomization while the TomoChain's randomization for block verifiers is potentially uniform and based on smart contracts. Furthermore, the use of voting as in TomoChain and DPoS enables a more incentive equality between stakers: In Ouroboros, stakers with very little stake have a very small probability of becoming block creators, while, in TomoChain, these stakers can choose an optimal strategy to vote for potential masternodes to get incentives.

V. CONCLUSION AND PERSPECTIVES

In this paper, we proposed PoSV, a PoS Voting-based blockchain protocol with heuristic and fair voting mechanism, rigorous security guarantees, and fast finality. We also presented a novel reward mechanism and show that, with this mechanism, the blockchain has a low probability of having forks, fast confirmation time, plus the contributions and benefits of masternodes are fair in the sense that the probability distribution function is uniform eventually.

Perspectives

- **Future work** The TomoChain team is currently working on the implementation of the Proof-of-Stake Voting, which will be released on schedule as stated in our roadmap. Furthermore, in parallel

17

with our novel consensus protocol, we will investigate the Sharding mechanism in order to provide even better transaction processing performance. We believe that, the Sharding technique with the stable number of masternodes will provide better stability and efficiency to the blockchain. At the same time, we commit to keep EVM-compatible smart contracts within our masternode sharding framework.

- **Economic sustainability** is also an important concept for a blockchain based decentralized network. That means to maintain the network in a sustainable condition, an equilibrium needs to be achieved, in which the cost of running the network infrastructure could be offset by the revenues generated. In this context, the cost of network infrastructure consists of two parts: the physical cost of having hardware such as servers, memories that passes the network technical requirements; and the capital cost of having TOMO locked into smart-contracts. The revenues for Masternodes would primarily come from Reward Engine emission, and later on from service revenues such as token exchange fees provided by applications running on top of TomoChain. We will publish a TomoChain economic analysis and proposal, separate from this technical paper in a later date.

REFERENCES

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronics cash system. 2008.

[2] Ethereum Foundation. Ethereum's White Paper. http://github.com/ethereum/wiki/white-paper, 2014. Online available 25/05/2018.

[3] D. Larimer. Delegated Proof-of-Stake (DPOS). BitShare White Paper 2014.

[4] S. King and S. Nadal. PPCoin: Peer-to-peer crypto-currency with proof-of-stake. Self-Published, 2012.

[5] V. Buterin. On public and private blockchains. Ethereum Blog, 2015.

[6] A. Kiayias, A. Russell, B. David, and R. Oliynykov: Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. IACR-CRYPTO-2017.

[7] D. Mingxiao, et al. A Review on Consensus Algorithms of Blockchain. 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC) Banff Center, Banff, Canada, October 5-8, 2017

[8] R. Pass and E. Shi. Rethinking Large-Scale Consensus. In the Proceedings of the IEEE 30th Computer Security Foundations Symposium, 2017.

[9] Thunder Token Foundation: Thunder Consensus White Paper, Janurary, 2018.

[10] R. Pass, L. Seeman, and A. Shelat. Analysis of the Blockchain Protocol in Asynchronous Networks. In EUROCRYPTO 2017.

[11] Juan A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In Elisabeth Oswald and Marc Fischlin, editors, Advances in Cryptology - EUROCRYPT 2015, Volume 9057 of Lecture Notes in Computer Science, pages 281–310. Springer, 2015.

[12] Tendermint Team. Understanding the Basics of a Proof-of-Stake Security Model. https://blog.cosmos.network/understanding-the-basics-of-a-proof-of-stake-security-model-de3b3e160710. Online available 25/05/2018.

[13] V. Buterin. On Settlement Finality. https://blog.ethereum.org/2016/05/09/on-settlement-finality/. Online available 25/05/2018.

[14] EOS Team. EOS.IO Technical White Paper v2. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md. Online available 25/05/2018.

[15] Bitshares Team. Delegated Proof-of-Stake Consensus. https://bitshares.org/technology/delegated-proof-of-stake-consensus/. Online available 25/05/2018.

[16] R. Pass, and E. Shi. (2017). Hybrid consensus: Efficient consensus in the permissionless model. In LIPIcs-Leibniz International Proceedings in Informatics (Vol. 91). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[17] V. Buterin, and V. Griffith. (2017). Casper the Friendly Finality Gadget. arXiv preprint arXiv:1710.09437.

[18] H. McCook. Under the Microscope: Economic and Environmental Costs of Bitcoin Mining. https://www.coindesk.com/microscope-economic-environmental-costs-bitcoin-mining/. Online available 25/05/2018.