Development of a Linear Scale Consensus Method of the Blockchain System

Gennady Shvachych, Boris Moroz, Maksym Khylko, Andrii Matviichuk, Vladyslava Kozenkova, and Volodymyr Busygin

Abstract The research highlights the development of a new and reliable blockchain system consensus method focused on applying a linearly scalable consensus mechanism. The new system is based on an analysis of available consensus mechanisms, sharding, and distributed randomness generation. The proposed approach allows the development of a blockchain with the following advantages: full scalability, security, energy efficiency, and fast consensus. Compared to known methods, the proposed shard option performs network connection and transaction verification and shows the current state of the blockchain. The decision threshold is low enough to allow small validators to participate in the network and receive appropriate rewards. The sharding process runs safely by applying a distributed randomness generation process that is unpredictable unbiased. In order to prevent adaptive Byzantine malicious validators, a continuous network reboot process is provided. The developed approach is resistant to fluctuations in the number of validators. It does not set a lower limit against the number of validators in every fragment, as in other solutions such as *Zilliqa*. Instead, an adaptive Proof-of-Stake-based model was adopted so that attackers can never occupy more than one-third of the shares that vote in a single shard that makes it reliable. The methods for creating the proposed blockchain improve available mechanisms for the blockchain functioning and have practical value for application in various digital economy sectors.

G. Shvachych
Ukrainian State University of Science and Technology, Dnipro, Ukraine

B. Moroz
Dnipro University of Technology, Dnipro, Ukraine

M. Khylko
National Academy of Science of Ukraine, Kyiv, Ukraine

A. Matviichuk
Vernadsky National Library of Ukraine, Kyiv, Ukraine

V. Kozenkova
State Agrarian and Economic University, Dnipro, Ukraine

V. Busygin (✉)
VUZF University (Higher School of Insurance and Finance), Sofia, Bulgaria
e-mail: busygin2009@gmail.com

G. Ranganathan et al. (eds.), *Pervasive Computing and Social Networking*, Lecture Notes in Networks and Systems 475, https://doi.org/10.1007/978-981-19-2840-6_16

Keywords Blockchain · Transactions · Stability · Security · Consensus mechanism · Reliability · Validation · Shard · Algorithm · Protocol · Distributed randomness

1 Introduction

Problem Statement. Blockchain technology is still under development. Therefore, the natural outcome is that the "correct" consensus protocol issue is still under discussion and debate. Many critical considerations, such as the degree of decentralization, underlie the spirit of blockchain as a technology. At least right now, there is no consensus on the "correct" consensus algorithm, which means that competition will only intensify in the future.

This paper proposes and explores a new blockchain system, operating on a linearly scalable consensus mechanism, a selection method, confirming a shard by voting shares, and scalable randomness generation using *VRF* (*Verifiable Random Function*) and *VDF* (*Verifiable Delay Function*) delays. The system is based on analyzing available consensus mechanisms, sharding, and distributed randomness generation. It is fully scalable, secure, energy-efficient, and with fast consensus.

Compared with available methods [1–4], the improved shard method performs network connection and transaction verification and exposes the state to the blockchain. The threshold is low enough for small validators to participate in the network and get rewards. The proposed sharding process occurs safely through a distributed randomness (*DRG*) process that is unpredictable, unbiased, and proven. The network is overloaded continuously to prevent slow adaptive byzantine malicious validators. Unlike other sharded blockchains requiring *Proof of Work* (*PoW*) model to select validators, the proposed consensus is based on *Proof of Stake* (*PoS*) and is therefore energy-efficient.

Proof of Stake (*PoS*) is a consensus model introduced in 2011 as an alternative to the *Proof of Work* (*PoW*) model. It aims to overcome the scalability limitations of *PoW* networks. Note that the goal of *PoW* and PoS is the same—to achieve consensus in the blockchain, however, the *PoS* model implements a different way of determining participants who verify transaction blocks. There are no miners on *PoS* blockchains. The participant priority according to the rules of the *PoS* algorithm does not depend on its computing power, but on the amount of cryptocurrency the participant possesses. Thus, the *PoS* model is devoid of the main disadvantages of the *Proof of Work* model:

- Mining consumes a huge amount of electricity power and hardware wears out quickly.
- The *Proof of Work* model provides a sufficient security level only if there is a large group of miners competing for block rewards. If the network is small a hacker can get a simple majority of the processing power and reorganize the blocks as he/she sees fit.

At the same time, validators, the cryptocurrency owners, support the *PoS* blockchain model functionality. They check user transactions, and if at least 2/3 of the validators agree that the transaction is correct, it is included in a new blockchain block. Simulteniously, in order to be eligible for block verification, participants need to block several coins in a specific blockchain smart contract. That process is known as staking. After that, the *PoS* protocol can choose a participant to validate the next block. Depending on the network, the choice may be random or according to the amount of cryptocurrency staked. As a reward, the selected validator receives a transaction fee from the verified block. As a rule, the more coins one blocks, the higher the chance of being selected.

Obviously, here the validators only do useful work (validation), and not numbers enumeration, so they do not have a race for performance, like miners. However, for the network to work efficiently and quickly, validators must run software on very powerful hardware, with a constant 24/7 network connection and a wide Internet channel.

Note that the main advantage of the *PoS* model is speed. Many *PoW* blockchains (such as *Bitcoin*) will never be able to process transactions as fast as *PoS* blockchains. And speed is a key factor for a network.

As a rule, the following arguments are given in *PoS* favor:

- significant resources to carry out an attack, which makes it impractical from a financial point of view;
- if the attacker has a large number of tokens, he/she will suffer from the attack, as that will violate the cryptocurrency stability.

Let us note the arguments that cause concern:

- *PoS* provides additional motivation for the resources accumulation in one hand, which may adversely affect the network decentralization;
- if a small group is formed that collects large enough resources, it can impose own rules for the network on other participants.

Thus, the *PoS* mechanism aims to improve blockchain technology and gain significant adoption in the industry. The lower barrier to entry into the mining process, no need for infrastructure, and relatively less vulnerability to attacks are currently attracting the community to have the *PoS*-based application and coins.

In the research in question, the consensus is achieved by a linearly scalable *Byzantine Fault Tolerance (BFT)* algorithm, faster than *Practical Byzantine Fault Tolerance* (*PBFT*) [5].

Analysis of Recent Research and Publications. The consensus protocol is a major element of a blockchain. It discovers how reliably and quickly the blockchain is verified to reach a consensus on the next block. Achieving consensus in a distributed environment comes down to solving the problem of Byzantine generals. The problem formulation is as follows. The Byzantine army includes n legions, each of which is commanded by its general, and the army also has a commander-in-chief, to whom the generals of the legions are subordinate. The army surrounds the city intending to attack. The favorable outcome of the war depends on the actions of all the generals, who need to communicate to come to a unified agreement on whether attack the city or not. However, there may be traitors among the generals, including the in-chief commander. A traitor can send orders of different content to different generals [6, 7]. Reaching consensus in such an environment is a quite difficult task. The problem's solution served as the basis for a mathematical model for the construction and development of blockchain technology. There is no central authority to provide for the same work on remote nodes in blockchain networks. Thus, there is a need for consensus protocols among distributed nodes. The crypto industry is not limited to applying well-known consensus mechanisms such as *PoW* or *PoS*. Researchers increasingly demonstrate new consensus protocols; in most cases, some modifications to the main available protocols. Let us consider some alternative mechanisms for achieving consensus in distributed environments.

The *BFT* algorithms are distributed network algorithms that allow consensus to be achieved even if some network nodes do not respond or give incorrect information. The *BFT* mechanism aims to protect against system failures by collective decision-making (both healthy and defective nodes) and reduce the impact of failed nodes.

The *Verifiable Byzantine Fault Tolerance* (*VBFT*) algorithm is a new consensus algorithm that combines traditional *PoS*, *Verifiable Random Function* (*VRF*), and *BFT*. This mechanism was developed specifically for the needs of the *ONTology* platform. *VBFT* can hold the consensus groups' scalability by *VRF*; it provides the randomness and fairness of the generation of the consensus set and ensures that the final state is reached quickly. In this algorithm, the centralization risk problem is eliminated. Conforming to the *ONTology*'s plan, the *VBFT* consensus algorithm could hold up to 2401 nodes onward while reaching consensus in <10 s [7]. The main advantage of such a protocol is the absence of the risk of centralization scalability, and hence it is resistant to attacks. The algorithm's disadvantage is that its application is limited by the *ONchain* company and the *ONTology* project.

A new class of consensus protocols was proposed to increase performance, the Byzantine Fault Tolerance (*PBFT*), wherein anonymity is not an essential aspect, i.e., nodes know some information about each other (initially nodes are authenticated). That allows consensus algorithms to be optimized with far higher throughput. The speed can increase tenfold, and data processing from hundreds to thousands of transactions per second, which is great for corporate realities.

This algorithm is used in the Hyperledger Fabric project as a consensus protocol, as it can process up to one-third of illegitimate transactions [8]. The protocol operation principle is as follows: the validator receives a message at the input, according to which the validator needs to decide whether to consider it true or not. To do this, the validator performs internal checks, followed by polling the other nodes one by one whether the transaction is valid. If two-thirds of the participants vote for this transaction as correct, the validator accepts it and transfers its decision to the blockchain network to other validators. It should also be noted that there is no hashing procedure in the *PBFT* algorithm [9].

Research Purpose. Based on the literature review and the analysis results of the current state of blockchain technologies development develop a linearly scalable blockchain consensus method. The new approach should be fully scalable, evidence-based secure, and energy-efficient blockchain; explore the functionality and features of the blockchain system based on next-generation sharding, which solves many problems of available blockchains; explore the stability and reliability of the developed blockchain system consensus method.

2 Statement of the Main Research Material

2.1 The Idea and Essence of the Blockchain System New Consensus Method

As the *PBFT* protocol improvement, the paper proposes a linearly scalable consensus mechanism for communication complexity. Instead of asking all validators to post their votes, the leader instigates the process of multi-signature signing to gather the validators' votes into $0(1)$—a multi-signature, and then relay it. Therefore, instead of receiving $0(N)$ signatures, every validator gets just one multi-signature, thereby reducing the communicating complexity with $0(\mathrm{N})^2$ to $0(\mathrm{N})$.

The idea of multi-signature $0(1)$ multi-signature improves the *BFT* method from the *ByzCoin* [10] blockchain, using the Schnorr signature scheme to aggregate constant size multi-valued signals and form a multicast tree between validators to further message transmission. Nevertheless, a multi-valued Schnorr signature requires a secret round of commitments, resulting in a total of two round-trip requests for a single multi-signature.

The paper proposes the method to improve available one by *BLS* (*Boneh-Lynn-Shacham*) multi-signature [11]; it requires only one round-trip request. Therefore, the method is designed to be at least 50% faster than the *BFT ByzCoin* method. Figure 1 presents the developed network communication method during one consensus round.

The developed method for conducting a consensus procedure involves the next steps:

1. The leader creates a brand new block and passes the header of the block to all validators. Conccurantly, the leader relays the block contents with erasure encoding (Fig. 1—the Announce phase).
2. The validators validate the header of the block, signs it with the *BLS* signature, and send the signature back to the leader (Fig. 1—Prepare phase).
3. The leader waits for at least $2f + 1$ valid signatures from validators (including leader's ones) and combines them into a *BLS* multi-signature. The leader then relays the aggregated multi-signature with the bitmap with the changes signed by the validators. Along with step 3, the *PBFT* "preparation" phase is completed.

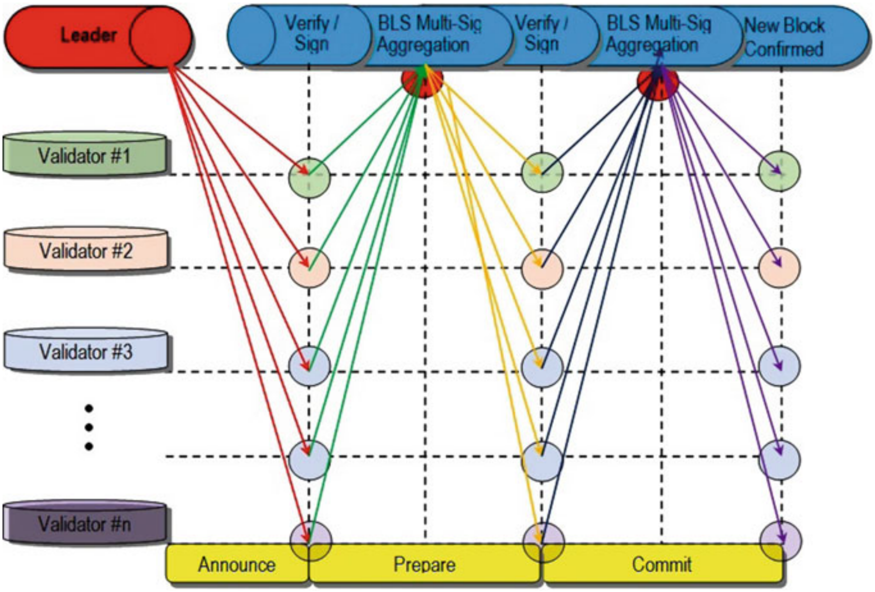

Fig. 1 Network communication of the developed method during one round of consensus (*Source* Authors' elaboration)

4. Validators check if it has at least multi-signature $2f + 1$ signers, verifies transactions with block content relays from the leader in step 1, signs the received message from step 3, and returns it to the leader.
5. The leader then awaits at least $2f + 1$ valid signatures and, starting at step 4, unites them into a *BLS* multi-signature creating a bitmap that registers everyone who signed. The leader then does a new block with all signed, multi-signed, and bitmaps and passes the new block to all validators. Along with step 5, the "fixation" phase of the *PBFT* ends (Fig. 1—the Commit phase).

Consensus validators are elected on a Proof-of-Stake basis. The proposed protocol differs from available *PBFT* wherein a validator with more voting shares has more votes instead of having one signature—one vote. Contrarary to waiting for $2f + 1$ signatures from validators, the leader awaits signatures from validators who collectively posses minimum $2f + 1$ voting shares.

Note that the common procedure to download the history of blockchain and reconstruct present state is extremely slow to allow re-introducing changes (it takes several days for the *Ethereum* blockchain to synchronize the history fully), given that the current state is much smaller than the entire history of the blockchain. To optimize the state synchronization process, making the blockchain state as small as possible is proposed. Loading the current state by epoch period is possible compared to loading the entire history.

In Ethereum, many accounts are empty and waste state-space on the blockchain. Empty accounts cannot be deleted due to possible replay errors when old transactions are resubmitted to a deleted account [12]. The problem can be solved by preventing replay attacks by allowing transactions to specify the current block hash: a transaction is valid only up to a certain number (e.g., 300) of blocks right after the block of the specified hash. Thus, old accounts can be deleted securely, which will greatly speed up the verification of the blockchain state.

New validators that join a shard initially download present state of that shard to enable a quick start validating transactions. The new node must make an appropriate check to ensure that the currently loaded state is valid. Instead of downloading the entire blockchain history to verify present state, the new node downloads the headers of the history block. It verifies them by affirming their signatures as cryptographic evidence (e.g. signatures and hash pointers) from present state to the genesis block. Signature verification is not computationally expensive, and it takes a significant time to verify all signatures onward from the genesis block. When it comes to mitigating this, the first block of each epoch is proposed to include an additional hash pointer to the first block of the last epoch. Thus, a new node can move over blocks during an epoch when it tracks hash pointers to the genesis block, which could greatly speed up checking the current state of the blockchain.

2.2 Stability and Reliability Research of the Developed Blockchain System Consensus Method

The proposed consensus method uses a different approach than the previously discussed one, with *Proof-of-Stake*, as a validator registration mechanism or a Sybil attack avoidanance mechanism. So that one may become a validator, prospective participants (or interested persons) must wager a certain number of coins to become eligible. The number of pledged tokens determines the number of voting shares destined for the validator. That method contains the main chain and many shards. The main chain serves as the ledger of identity, while the shard chains store the blockchain individual states and simultaneously process transactions. That algorithm uses randomness generation by combining a *VRF* and a *VDF* and incorporates *PoS* into the sharding process, shifting fragment protection concerns from a minimum number of nodes to a minimum number of voting shares.

Each voting particle represents one vote in the *BFT* consensus. Stackers receive voting shares proportional to their tokens. Voting shares are further taken up sharding randomly. Figure 2 depicts the sharding procedure interpretation by voting shares. Tokens become validators for the fragments they vote for.

A voting share is a virtual ticket allowing a validator to cast one consensus vote. Validators can purchase voting shares by betting tokens. The number of tokens required to vote is algorithmically adjusted. At the start of each phase, new shares with validator voting rights will be taken up to shards randomly. New validators join the shard where voting shares are assigned to them. Consensus in a given fragment is achieved by validators who, at least, must jointly have $2f + 1$ voting shares to sign

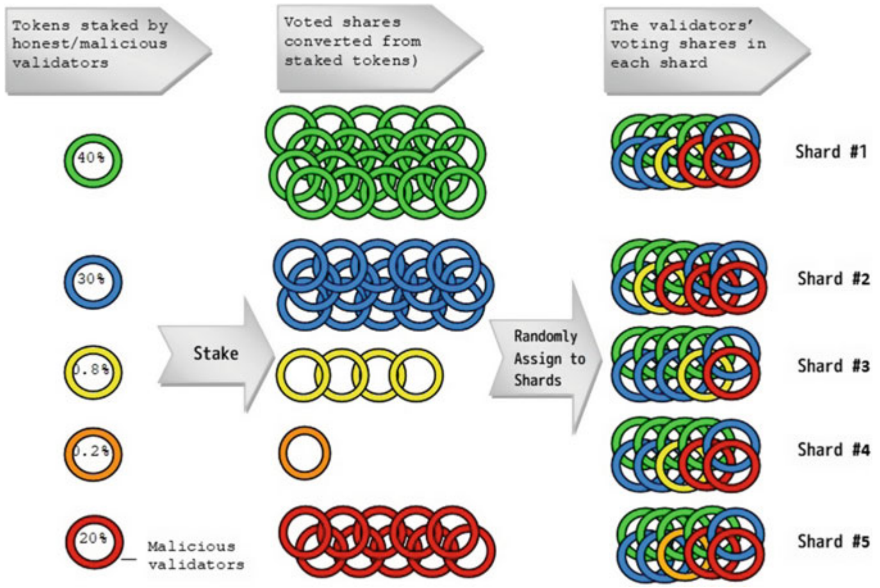
Tokens staked by honest/malicious validators
Voted shares converted from staked tokens)
The validators' voting shares in each shard
40%
30%
0.8%
0.2%
20%
Malicious validators
Stake
Randomly Assign to Shards
Shard #1
Shard #2
Shard #3
Shard #4
Shard #5

Fig. 2 Sharding procedure via shares voting (*Source* Authors' elaboration)

the block. To guarantee the safety of one shard, the number of voting shares in malicious validators needs to be less than one-third of all voting shares of this shard. The adaptive *PoS* of the proposed consensus method guarantees security requirements by adaptively adjusting the share price voting rights and taking up individual voting shares to shards instead of individual auditors.

The sharding security by voting shares is that even if one-fourth of all pledged tokens are malicious validators, one shard is assigned to one validator. Thus, the worst, when one malicious validator holds all the tokens (voting shares), less than one-third of the voting tokens in this shard. The reason is that the rates for each shard are m times less than the rates for the entire network, where m is shard's number. Thus, preventing the attack of malicious validators instead of sharding by validators, voting shares are split (one share has the right to vote, one shard is assigned).

In this method, the share price has a vote is set in algorithmical order so it is small enough to prevent malicious participants concentrate their voting power in one shard. The share price has the right to vote is set in tokens P_{vs}:

$$P_{vs} = \frac{Ts_{e-1}}{N_{sh} \cdot \lambda} \tag{1}$$

wherein: λ is security parameter;
N_{sh} is the shard number;
Ts_{e-1} is a total number of phase tokens $e - 1$.

Let us prove that when λ > 600, the probability of failure is negligible. In this case, the chance of a single shard should have over one third of the malicious voting shares.

Given the definition P_{vs}, the total number of voting shares is:

$$N_{sh} = \frac{Ts_{e-1}}{\lambda \cdot P_{vs}} \tag{2}$$

Given the credible source of randomness and the randomness-based sharding process, the probability distribution of the malicious voting shares in every part can be represented as a hypergeometric distribution (random sampling without replacement).

$$P(X = k) = \frac{C_n^N - C_k^K}{C_n^N} \tag{3}$$

wherein N is overall qauntity of voting shares;
C_n^N is the number of combinations of N to n;
N is total number of voting shares;
$K = \frac{N}{4}$ is maximum qauntity of malicious voting shares;
$n = \frac{N}{N_{sh}}$ is the qauntity of voting shares in every shard;
k is the qauntity of malicious shares with voting rights in the shard.

Actual shard bounce rate P $(X \le k)$ follows from the cumulative hypergeometric distribution (N, K, n, k), which is in case if N is large, reduses to a binomial distribution (i.e., random sample with replacement):

$$P(X \le k) = \sum_{i=0}^{k} n_i \cdot p_i \cdot (1 - p)^{n-i} \tag{4}$$

When n is high enough, the shard's probability that contains more than one third of malicious tokens is negligible.

When $n = 600$, the probability that the shard contains more than one third of malicious shares with voting rights P $(X \le 200) = 0{,}999{,}997$, which shows the failure of such a shard, i.e., no consensus can be reached.

To ensure high security of the shard λ = 600. Parameter λ regulates the minimum number of voting shares that one shard must contain.

This solution is functional like the least quantity of nodes in a shard, stated in other *PoW*-based solutions [13–15]. This approach is robust against fluctuations in the number of validators. It does not set a lower limit on quantity of validators in each part like other solutions, e.g. *Zilliqa*. On the other hand, an adaptive *PoS*-based model was adopted so that attackers could never hold over one-third of the voting shares in one shard, making it reliable.

Peculiarity analysis of scalable randomness generation by *VRF* and *VDF* functions. The approach to the new blockchain formation developed in the thesis combines the advantages of the considered solutions. The proposed method uses the *BFT* consensus to ensure the random number finality. In particular, the protocol covers next steps:

1. The leader transmits a message to all validators with the last block hash H (B_{n-1}).
2. After receiving the message, the *VRF* is computed for every i validator to generate a random number r_i, and the proof p_i: $(r_i, p_i) = VRF\ (sk_i, H\ (B_{n\text{-}1}))$, v, wherein sk_i is the validator's secret key i,v is the current consensus number. Followed by each validator returning (r_i, p_i) to the leader.
3. The leader waits until he gets at least $f + 1$ real random numbers and combines them to get the preimage of *pRnd* ultimate randomness.
4. Leader provides *BFT* consensus of all validators to get consensus on *pRnd* and fix it to the B_n block.
5. After *pRnd* is executed, the leader begins to compute the actual probability *pRnd* = *VDF (pRnd, T),* where *T* is the *VDF* complexity and is set algorithmically so that the probability can be computed only after *k* blocks. When the computation *pRnd* is in progress, the leader initiates *BFT* among all validators to reconcile reality *pRnd* and generate a probability on the blockchain (Fig. 3).

Thus, the *VDF* function is used to delay the expansion of *Rnd* evidentially and to prevent the malicious leader from preempting randomness by choosing a value for a subset of the *VRF* random numbers.

Owing to the *VDF* function, the leader cannot know the actual final randomness of *pRnd* until it is added to the blockchain. While *VDF* evaluates rnd, *pRnd* was sent to the previous block, so the leader can no longer manipulate it.

The worst that a malicious leader can do is either to add randomness to *pRnd*, or stop the protocol without fixing *pRnd*. That does not harm as the waiting mechanism switches the leader and restarts the protocol. In the long-term future, there could be invented *ASIC*s (application-specific integrated circuits) to compute a *VDF* function that can find vulnerable nodes and compute result ahead other true nodes. However, at present, such robust circuits are not invented.

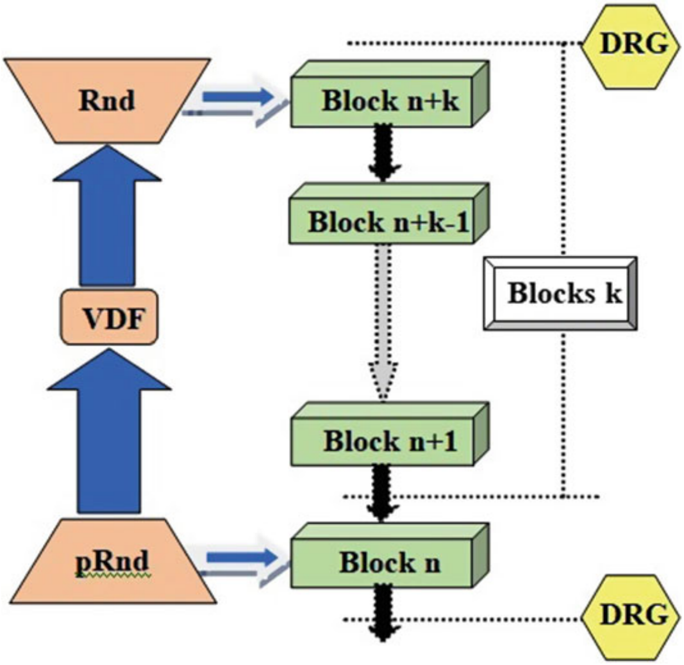

Fig. 3 A mechanism for delaying the final randomness detection by verifiable delay function (*Source* Authors' elaboration)

3 Conclusions

This paper's research is devoted to the development of a new and reliable blockchain system consensus method focused on the linearly scalable consensus mechanism. The proposed approach is based on the analysis of available consensus mechanisms, sharding and generation of distributed randomness. The proposed consensus mechanisms allow the development of a blockchain with the following advantages: full scalability, security, energy-efficienct, and with fast consensus.

The proposed consensus method improves available ones by *BLS* (*Boneh-Lynn-Shacham*) multi-signature. However, it requires only one round-trip request. In this regard, the developed method is at least 50% faster than the *ByzCoin BFT* method. The paper presents an algorithm for conducting the consensus procedure. Consensus validators are elected based on an adaptive *Proof of Stake* model.

The proposed protocol differs from available *PBFT* in that a validator with more voting shares has more votes than others rather than one signature one vote. In order to become a validator, prospective participants (or interested parties) must stake a certain number of tokens to become eligible. The number of pledged tokens determines the number of voting shares. That method has the main chain and many shards. The main chain serves as the ledger of identity, while the shard chains store the individual states of the blockchain and simultaneously process transactions. That algorithm uses randomness generation by combining *Verifiable Random Function* and *Verifiable Delay Function* and incorporates a *PoS* model into the sharding process that shifts fragment protection concerns from minimum nodes to a minimum voting shares. The number of tokens required to vote is algorithmically adjusted. At the start of each phase, for new validator voting shares there will be randomly assigned shards.

To guarantee the security of one shard, the number of voting shares in malicious validators needs to be less than one third of all voting shares of this shard. The adaptive *PoS* of the proposed consensus method guarantees security requirements by adaptively adjusting the share's price, has the right to vote, and assigns individual voting shares to shards rather than individual verifiers.

The sharding security by voting shares lies in the fact that even if for all pledged tokens 1/4 are harmful validators, then one shard is assigned to one validator. Then in the worst case, where a single malicious validator holds all the tokens (voting shares), it will have less than one third of the voting tokens in that shard.

To ensure high shard security, the network security parameter regulates the minimum voting shares that one shard must hold. Such a solution functionally corresponds to the minimum nodes in a certain network, described in some other solutions based on the *PoW* model. Thus, the presented approach is resistant to fluctuations in the number of validators. Moreover, it does not set a lower limit on the number of validators in each fragment, as in other solutions such as *Zilliqa*. Instead, an adaptive *PoS*-based model was adopted so that attackers can never occupy over one third of the voting shares in one shard, which makes it reliable.

The methods for creating the proposed blockchain improve available mechanisms for the functioning of the blockchain and have practical value for use in various digital economy sectors.

References

1. Centralized Decision Making Helps Kill Bad Products (2021) Harvard Business Review, August 31. https://hbr.org/2016/10/centralized-decision-making-helps-kill-bad-products
2. Krupskyi OP (2014) Modern management decision-making methods and their connection with the organizational culture of the tourism enterprises in Ukraine. Econ Ann-XXI 1(7–8):95–98
3. Campbell A, Kunisch S, Müller-Stewens G (2021) To centralize or not to centralize? McKinsey & Company, March 1. https://www.mckinsey.com/business-functions/people-and-organizational-performance/our-insights/to-centralize-or-not-to-centralize
4. Zavorotny A (2018) Analysis of practice of blockchain technology application in financial management. Politech Stud J 27. https://doi.org/10.18698/2541-8009-2018-10-391
5. Treiblmaier H (2019) Toward more rigorous blockchain research: recommendations for writing blockchain case studies. Front Blockchain 2. https://doi.org/10.3389/fbloc.2019.00003
6. Javed MU, Rehman M, Javaid N, Aldegheishem A, Alrajeh N, Tahir M (2020) Blockchain-based secure data storage for distributed vehicular networks. Appl Sci 10(6):2011. https://doi.org/10.3390/app10062011
7. Nick A, Hoenig L (2020) Consensus mechanisms in blockchain technology. Lexology, May 7. https://www.lexology.com/library/detail.aspx?g=e30e7d54-3c7f-4ca0-8a22-478227a9b5ec

8. The Truth about Blockchain (2019). Harvard Business Review, August 21. https://hbr.org/2017/01/the-truth-about-blockchain
9. Cain D (2019) Big data synchronization: 5 ways to ensure big data accuracy. Medium, July 10. https://towardsdatascience.com/big-data-synchronization-5-ways-to-ensure-big-data-accuracy-4c4801b021ad
10. Gimenez-Aguilar M, de Fuentes JM, Gonzalez-Manzano L, Arroyo D (2021) Achieving cybersecurity in blockchain-based systems: a survey. Futur Gener Comput Syst 124:91–118. https://doi.org/10.1016/j.future.2021.05.007
11. Tyagi N (2020) Top 10 big data technologies|analytics steps. Analyticssteps.com. https://www.analyticssteps.com/blogs/top-10-big-data-technologies-2020
12. Gour R (2019) Big data architecture—the art of handling big data. Medium, September 18. https://towardsdatascience.com/big-data-architecture-the-art-of-handling-big-data-bc565c3a7295
13. Wroughton J, Cole T (2013) Distinguishing between binomial, hypergeometric and negative binomial distributions. J Stat Educ 21(1). https://doi.org/10.1080/10691898.2013.11889663
14. Shvachych G, Busygin V, Zaporozhchenko O, Sazonova M (2020) Some aspects of the practical implementation of the blockchain technology. In: Savchuk L, Bandorinaya L (eds) Monograph: state, industries, enterprises, business: realities and trends of economic, informational and technical development. Porogy, pp 284–304
15. Shvachych G, Ivanov R, Busygin V (2019) Blockchain technology as a means of improving enterprise efficiency. In Shebeko K (ed) Collection of scientific papers of the tenth international scientific and practical conference on banking economics, October 2019. PolesGU, pp 364–368