Journal of Engineering Science — Vol. XXIX, no. 1 (2022), pp. 97 - 104
Fascicle — Electronics and Computer Science — ISSN 2587-3474
Topic — Computers and Information Technology — eISSN 2587-3482

https://doi.org/10.52326/jes.utm.2022.29(1).08
CZU 004

PROOF-OF-STAKE CONSENSUS ALGORITHMS FOR THE SOFTWARE COMPONENTS BLOCKCHAIN

Vaidas Giedrimas*, ORCID: 0000-0003-1979-9978

Vilnius University, Siauliai Academy, 84 Vytauto St., LT-76352 Siauliai, Lithuania
*Corresponding author: Vaidas Giedrimas, *vaidas.giedrimas@sa.vu.lt*

Received: 11. 05. 2021
Accepted: 01. 19. 2022

Abstract. In the blockchain context, the information system (IS) is considered a part of its infrastructure. However, blockchain itself can be used for IS development using software components and services. As the trust for binary components or services is still a problem, we propose to use the blockchain of components to solve this problem. In this paper, the part of such solution, namely consensus algorithms, is discussed. We focus on Proof-of-Stake algorithms and present their feasibility to be used in the blockchain of software components. It was found that the use of probabilistic algorithms (RRR, CloudPoS, WV, DDPoS, Panda) allow the partial solution of the problem in the blockchain of reliable software components.

Keywords: *software component, CBSE, consensus algorithm, blockchain.*

Abstract. În contextul blockchain, sistemul informational (SI) este considerat o parte a infrastructurii sale. Cu toate acestea, blockchain-ul în sine poate fi folosit pentru dezvoltarea SI folosind componente și servicii software. Deoarece încrederea pentru componentele sau serviciile binare este încă o problemă, ne propunem să folosim blockchain-ul de componente pentru a rezolva această problemă. În această lucrare este discutată o astfel de soluție, și anume algoritmii de consens. Ne concentrăm pe algoritmii Proof-of-Stake și prezentăm fezabilitatea acestora pentru a fi utilizați în blockchain-ul componentelor software. S-a constat, că utilizarea algoritmilor cu finalitate probabilistică (RRR, CloudPoS, WV, DDPoS, Panda) permite rezolvarea parțială a problemei în blockchain-ul componentelor software de încredere.

Cuvinte cheie: *componentă software, CBSE, algoritm de consens, blockchain.*

Introduction

Distributed software in most cases consist of various distributed components or is made using a few orchestrated software services. As the software components are binary, they can be the objects of the commerce (selling and buying) and the objects of third-part composition. It is possible to reuse the functionality of some component without any idea what exactly its implementation is. In the other hand, system integrators cannot to evaluate component in other ways than the black box approaches. In result they cannot be sure that the component is not harmful. The problem of the lack of trust on software components is already discussed in various sources [4, 11, 13, 22, 24, 25, 30]. There are few possible solutions to increase trust on the component:

- Trusted computer security evaluation criteria (TCSEC) by NSA and NIST, USA;
- Trusted Components Initiative (TCI);
- The use of various formal methods for ensuring *a-priori* correctness of the components (e.g., type theories, category theories, mathematical frameworks) [15, 31];
- Prediction-oriented component-based technologies, for the analysis of the relationship between structural restrictions and assumptions of the domain models, which helps to predict properties of the systems [13].

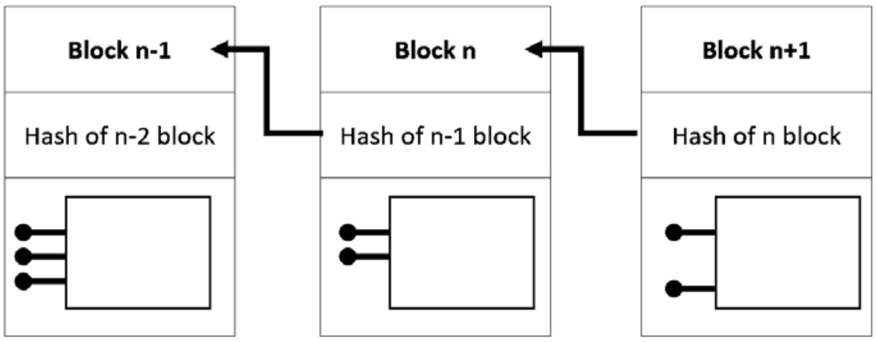

Figure 1. Blockchain of the components.

The methods can be divided to two categories: invasive methods and non-invasive ones. Invasive methods are used for such purposes as component stress testing, prediction of component inner properties or to make simulation of real-time use of the component. If invasive methods are used, then the component is considered as a “gray-box”, or “glass-box”. Sometimes even reverse engineering methods are used to reveal an internal structure of the component, or the algorithms used in its implementation.

Non-invasive methods are based on hash codes, formal certificates and so on. The component is considered as “safe” and “trustful” in the case well-known manufacturer, or some another authority approve it. From some point of view the situation is comparable to the banking system. E.g., if the Government of some state claims that its coins and banknotes are trustful, we consider this currency as reliable. Consequently, non-invasive trust of software components have the same disadvantages as the banking system have:

- The power is not in the user-side. Well-known manufacturer of the components or an open-source community can cancel the support as easy as the governments can devalue its currency. The user/developer is only informed in the best case, but no negotiations are made before.
- The threat to be hijacked. Some hackers can try to hijack well-known component or made fake signatures of the components (on behalf of real manufacturer) as the criminals are trying to make fake banknotes or goods.

Nowadays the banking system is started to move towards big “promise” of FinTech age, i.e., towards decentralization [20]. Trustful, DLT-based systems are among the contemporary hype cycle [17]. DLT or blockchain is considered as a storage point for the descriptors of software components in this paper. Here and below, we will call it as the “blockchain of the components”. This blockchain consists of the records, with the comprehensive information about the component interface, functionality, manufacturer, and rating (see. Figure 1). For the security reasons binary component (or trustful link to it), described in the blockchain node, is stored in the same record as well. The goal of the blockchain is to increase the trust in software components. As pointed in [16], such secure structure has a promising side effect – distributed component repository.

Every blockchain is based on some Consensus algorithm, which provides the rule which exactly node should be able to produce a record and how the changes must be made. Most of the algorithms is tuned to work in potentially dangerous conditions. It is known that some nodes in the distributed system can be malicious/unfair. But the algorithm must ensure that the right decision would be made event in that case. The goal of this paper is to select blockchain consensus protocols, which could be used in the blockchain of trustful software components (or services). The rest of the paper is organized as follows: Section 2 shortly introduces related work in this area, Section 3 presents the consensus algorithms and their selection criteria. Section 4 outlines the consensus achieving algorithms of Blockchain of components (or services). Finally, the conclusions are made, and some consecutive questions are exposed.

Related work

It is several references about the use of the blockchain for different purposes: academic affairs in the university [18], e-voting [5], medical image sharing, supply chain management, software engineering [10, 12, 14, 21, 27] etc. Despite it, we almost not found the papers covering the use of the blockchain for component-based software engineering nor about the component trust solutions using DLT.

The paper [33] is dedicated to the support of operations, but not to ensure the trust on software assets. The topic is introduced in [16], where the overall possibility for using blockchain as a helper for CBSE is discussed. According to [16], Proof-of-work (PoW) and Proof-of-authority (PoA) consensus algorithms are not suitable to use in component-based software context. The former is not applicable, because is nothing to “burn” in this context. The usage of the later (PoA) is not desirable as it would led back to centralization [16].

Selection criteria

There is a high number of blockchain consensus algorithms having different names and (partially) different aspects [7]. However, for simplicity reasons, we will focus on a very comprehensive study [8], which claims that most of the existing blockchain consensus algorithms by their nature (or origin) can be divided into 3 groups: Proof-of-Stake (PoS), Proof-of-Work (PoW), and Practical Byzantine Fault Tolerance (PBFT).

In PoW, the right to write the block is assigned to the node, which just solved dedicated crypto problem. It is a basis on most of cryptocurrencies. In PoS, the block can be created by the node selected in random order. However in the set of “lottery”-participants could be only the nodes, which can put at “stake” (i.e. to use some part of the goods/cryptocurrency/time/etc.). The block is approved if the confirmations from at least 51% nodes are got. In PBFT [8, 9] three phases are used for decision: initial, prepare and commit. In initial phase candidate node send the block to backup nodes (via primary node). If the conditions for the validity of the block is met, then the backups pre-accept this block. Then, in preparation phase the block is considered by all the nodes (not only the backups).

We continue the work of [16], in which 10 possible consensus algorithms were considered: PoW, PoS, DPos/LPos, Proof of Burn (PoB), Proof of Authority (PoA), Ripple Protocol Consensus Algorithm (RPCA), Stellar Consensus Protocol (SCP), Register-Deposit-Vote (RDV) [26], and Hashgraph [1, 2, 28]. As pointed in [16], PoW, PoB, and few other algorithms are not suitable for the blockchain of components, so we will focus only on the PoS algorithm class in this paper.

Next, we will examine each algorithm from the PoS class, using the classification framework, described in [8]. The framework consists of 4 categories: origin, design, performance, and security. The assessment will be made having in mind two contexts: a) the blockchain of the components and b) blockchain of the services.

The evaluation of POS consensus algorithms

From the list consisting of almost 30 algorithms [8] by the criteria of origin, we selected 7 consensus algorithms based on PoS:

- Medical image sharing (MIS) algorithm is proposed for the healthcare domain. Using this algorithm, the probability of a node producing the next block is driven by the number of Define Study transactions designating that node's public key as the image source. However, the procedure of block validation is not explained in more detail.
- Robust RR (RRR). Round Robin theory-based algorithm is specific by using deterministic round-robin selection with a lightweight leader endorsement mechanism. Every time the set of oldest identities are chosen as leading candidates. Endorsers are selected randomly form the set of active identities. A candidate proposes a block and the endorsers confirm the block from the oldest candidate they observe. The leader candidate (who receives the required number of confirmations from the endorsers), is chosen as the leader to add a new block. The authors of RRR [3] claim that the algorithm guarantees the highest possible probability, that only one block from one leader is produced on each round and that re-writing cannot be made.
- Fantomette. This protocol uses BlockDAG theory and is based on Caucus [6, 8]. It is a secure leader election protocol, providing game-theoretic guarantees. The participants (nodes) commit its states to a public list and then the nodes are considered eligible to be elected leader after some fixed number of rounds. In the first round, when some number of participants is reached, participants run a secure coin-tossing protocol to obtain a random value. If each next round, every participant verifies their eligibility. The eligible participant (if any) creates a transaction with its data and broadcasts it to their peers.
- CloudPoS. In this algorithm, the concept of epoch is used. For each epoch, the need to stake resources once again to be electable as the leader (for this epoch). After the bets, the leader is selected randomly based on the individual stakes. Only the leader can create a block, however, this must be approved by most participants [8, 29].
- Trust-CP. The Trust Consensus Protocol [23] introduces the concept of trust score. Trust score is calculated for each participant. The scores must be committed using separate blockchain transactions (as the blocks are committed for the writing). When the block is committed, then the trust score of the "block producer" must be recorded in the same block as well. The goal of Trust-CP is a blockchain-based solution to ensure trust between peers (participants).
- Weighted Voting (WV). This is an extension of PoS, solving the problem of non-active validators [19]. If many validators would abstain from the voting, in classic PoS, the block would be not confirmed and written, even though the majority of

voted participants would approve it. In WV the validators have the weights of votes, so the problem would persist only in case if some "heavyweight" node does not vote. In another hand, this algorithm is more near to centralized solutions.
- BIFTS. The blockchain-IoT-based food traceability system [8] is a variation of the PoSCS consensus algorithm. The main contribution in BIFTS is an application area (supply chain management, as in Proof-of-Supply-Chain-Share (PoSCS) protocol), not the technical novelty.

And 2 more algorithms (partially) based on Delegated PoS (DPoS) were selected as well:
- DDPoS. Delegated Proof-of-Stake with Downgrade mechanism algorithm [34] is proposed as a solution for high centralization in other consensus algorithms, lack of security, and efficiency of generating blocks. In DDPoS, there are 101 nodes with the most votes that can be elected as witness nodes. This shortlists as in DPoS, is the subject of a regular update. During the update the nodes with the low rate of generating blocks and/or malicious behavior are removed, loses their credibility and witness identity. The consensus process of the DDPoS algorithm consists of three modules: selecting a certain number of consensus nodes, reaching a consensus on the verification of the block, and downgrading malicious nodes [34].
- Panda, or DPoS-BA-DAG consensus algorithm [35] is used in DLattice blockchain. The candidate secretly generates the consensus identity locally, and the consensus committee for this branch is also made at the same time. The algorithm is divided into two phases: voting and commit. Initially, the dedicated members of the voting committee select a TB to vote, and all consensus nodes get the vote results. In the second phase, the members of committees is starting to commit voting. The commit is done using previuosely collected vote results. IN result all the nodes get the commit voting results. The consensus is considered as done when the node counts commit voting results exceeds the threshold value.

By the Design category from Bouraga's classification framework [8], we evaluated RRR, Fantomette, CloudPoS, Trust-CP, WV, DDPoS, and Panda algorithms. RRR and CloudPoS use 3rd party tools for permission management, Trust-CP - for the trust evaluation. The other 4 approaches are self-contained. Most of the approaches use the reward as the main incentive for the nodes, while only DDPoS uses punishment. In the CBSE world, a punishment could be out-of-the-stock action, especially in the components-of-the-shield (COTS) case. The reward can be various, e.g., the higher rating in the software component list, increased level of trust, and so on. The concept of the stake itself is in the context of CBSE, can be defined as one of the follows:
- The number of already recorded component descriptions in the blockchain.
- The rate of recorded component descriptions. For example, if a new record about a new component in blockchain was rejected, that could mean that its quality is below the voters' expectations. If it happens in the second or n-th case, then the trust in existing products (software components) of the same manufacturer could be decreased.
- The (decreasing) number of possible submissions for recording in the future.

In the blockchain of software components, the concept of finality is controversial. If the finality is deterministic (i.e., it is impossible to add a node before already recorded one) the trust for the component. As pointed in [16], in component-based software engineering, the presence of different versions of the components is vital. However, in the blockchain of components only some version (or some vendor) of the component will be marked as "trustful". All consecutive versions must to get new approval from all the nodes. In other hand, in the case of the adoption of a consensus algorithm with probabilistic finality, the problem of component versioning can be (partially) solved. Probabilistic finality is observed in RRR, CloudPoS, WV, DDPoS, and Panda algorithms.

Almost all the 7 algorithms have a formal background, except for the Trust-CP which formal development was not found in [23].

The Performance category (3rd in the Bouraga's classification framework [8]), is less relevant in the blockchain of the components unless we would talk about continuous development. Usually, the process of submission of a new component to the repository takes some time. e.g. the period of approving a new app for Android in GooglePlay repository can be done from few hours to 2 weeks (in the moment of this paper writing). By the number of successful transactions per second starting from the first transaction deployment time the best algorithms are [8] RRR (1500) and Panda (1200). Most of the algorithms do not have any latency between the completion time and the deployment time, however using CloudPoS the gap in time could be 10-15ms, while RRR - a minute. Unfortunately, the fault tolerance is not on the spot of the selected 7 algorithms. At least this wasn't considered in the referred literature. Only Trust-CP reports that in 84% cases the algorithm can recover from the failure of a node. Unfortunately, only RRR algorithm supports the scalability. Other algorithms should be improved according to these criteria. All seven algorithms are proved by its authors in some experimental evaluation.

The Security category [8], is very important as we focus on trust in components in the blockchain. They are reported thousands of cases of malware in information systems, which imply big organizational, performance, trust, and other problems. The problems would even bigger if the malware would be recorded to the Trusted component repository (in our case the blockchain of components). We consider 7 algorithms by their resilience to 4 most frequent threats [8, 32]: Sybil, DDoS, "51 percent", and Eclipse attacks. Unfortunately, no one algorithm from our research is ready for the Eclipse attack (when intruders influent the network that any communication must be performed with malicious nodes only). However other types of attacks are not so vulnerable for the selected consensus algorithms. Sybil attacks can be handled by Fantomette and Trust-CP algorithms, DDoS - by Fantomette, 51% attack - by RRR, CloudPoS. WV and DDPoS seams are the least secure variations of consensus algorithms. The best security level showed Panda, as it could resist Sybil, DDoS, and "51 percent" attacks.

Conclusions and future work

After the assessment of the Prove-of-the-Stake consensus algorithms and their possible implementation in distributed software components trust-ensurance system, we came to the conclusions:

1. Comprehensive classification frameworks for consensus algorithms exists, however the idea of using blockchain for component-based software engineering still have week coverage in scientific references.
2. The concept of the Stake using PoS algorithms in CBSE context could be defined as: a) the number of already recorded components, b) the rate of recorded components, c) the number of possible submissions for recording in the future. However, this concept should be discussed in more details in future.

3. The use of the algorithms with probabilistic finality (RRR, CloudPoS, WV, DDPoS, Panda) enable to (partially) solve the problem of component versioning in the blockchain of trusted software components.

Acknowledgements. This paper was presented at the *International Conference on Electronics, Communications, and Computing, IC ECCO*, 21-22 October, 2021, Chisinau, UTM.

References

1. Hashgraph consensus: Detailed examples. Tech. Rep. SWIRLDS-TR-2016-02, Swirlds, Inc. (2016), [online]. [accesat 10.04.2021]. Disponibil: http://leemon.com/papers/2016b2.pdf
2. The swirlds hashgraph consensus algorithm: fair, fast, byzantine fault tolerance. Tech. Rep. SWIRLDS-TR-2016-01, Swirlds, Inc. (2016), [online]. [accesat 10.04.2021]. Disponibil: http://leemon.com/papers/2016b.pdf
3. Ahmed-Rengers M., Kostiainen K. Don't mine, wait in line: Fair and efficient blockchain consensus with robust round robin, 2020.
4. Alvaro A., Land R., Crnkovic I. Software component evaluation: A theoretical study on component selection and certification. Tech. Rep. ISSN 1404-3041 ISRN MDH-MRTC-217/2007-1-SE, November 2007.
5. Awalu I.L., Kook P.H., Lim J.S. Development of a distributed blockchain evoting system. In: *Proceedings of the 2019 10th International Conference on e-Business, Management and Economics*. pp. 207–216. ICEME 2019, Association for Computing
6. Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3345035.3345080
7. Azouvi S., Mccorry P., Meiklejohn S. Betting on blockchain consensus with fantomette (2018)
8. Bach L.M., Mihaljevic B., Zagar M. Comparative analysis of blockchain andnsus algorithms. In: *Proceedings of 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. pp. 1545–1550, 2018. https://doi.org/10.23919/MIPRO.2018.8400278
9. Bouraga S. A taxonomy of blockchain consensus protocols: A survey and classification framework. *Expert Systems with Applications* 168, 114384, 2021. [online]. [accesat 01.03.2021]. Disponibil: https://doi.org/https://doi.org/10.1016/j.eswa.2020.114384,
10. Castro M., Liskov B. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.* 20(4), pp. 398 – 461, 2002.
11. Chakraborty P., Shahriyar R., Iqbal A., Bosu A. Understanding the software development practices of blockchain projects: A survey. In: *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. pp. 28:1–28:10. ESEM '18, ACM, New York, NY, USA, 2018.
12. Chaudhari D., Zulkernine M., Weldemariam K. Towards a ranking framework for software components. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. pp. 495 – 498. SAC '13, ACM, New York, NY, USA, 2013.
13. Chawla C. Trust in blockchains: Algorithmic and organizational. *Journal of Business Venturing Insights 14*, e00203, 2020, [online]. [accesat 05.01.2021]. Disponibil : https://doi.org/https://doi.org/10.1016/j.jbvi.2020.e00203.
14. Crnkovic I. *Building Reliable Component-Based Software Systems*. Artech House, Inc., Norwood, MA, USA, 2002.
15. Destefanis G., Marchesi M., Ortu M., Tonelli R., Bracciali A., Hierons R. Smart contracts vulnerabilities: a call for blockchain software engineering? In: *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. pp. 19 – 25, 2018.
16. Dimitriou T., Michalas A. Multi-party trust computation in decentralized environments. In: *2012 5th International Conference on New Technologies, Mobility and Security (NTMS)*. pp. 1 – 5, 2012. https://doi.org/10.1109/NTMS.2012.6208686
17. Giedrimas V. The role of blockchain for increase trust on software components and services. In: *2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)*. pp. 1–4, 2020. https://doi.org/10.1109/AICT50176.2020.9368582
18. Hawlitschek F., Notheisen B., Teubner T. The limits of trust-free systems: A literature review on blockchain technology and trust in the sharing economy. *Electronic Commerce Research and Applications 29*, pp. 50-63, 2018.
19. Ismail L., Hameed H., Alshamsi M., Alhammadi M., Aldhanhani N. Towards a blockchain deployment at uae university: Performance evaluation and blockchain taxonomy. In: *Proceedings of the 2019 International Conference on Blockchain Technology* (*ICBCT 2019)*. pp. 30 – 38., Association for Computing Machinery, New York, NY, USA, 2019.

20. Leonardos S., Reijsbergen D., Piliouras G. Weighted voting on the blockchain: Improving consensus in proof of stake protocols. In: *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. pp. 376 – 384, 2019.
21. Nicoletti B. *The Future of FinTech: Integrating Finance and Technology in Financial Services*. Palgrave Macmillan, 2017.
22. Porru S., Pinna A., Marchesi M., Tonelli R. Blockchain-oriented software engineering: Challenges and new directions. In: Proceedings of the 39th International Conference on Software Engineering Companion. pp. 169–171. ICSE-C '17, IEEE Press, Piscataway, NJ, USA, 2017.
23. Roshandel R. E. A. Estimating software component reliability by leveraging architectural models. In: *Proceedings of the 28th International Conference on Software andeering*. pp. 853–856. ICSE '06, ACM, New York, NY, USA, 2006.
24. Shala B., Trick U., Lehmann A., Ghita B., Shiaeles S. Novel trust consensus protocol and blockchain-based trust evaluation system for m2m application services. *Internet of Things 7*, 100058, 2019. [online]. [accesat 14.12.2020]. Disponibil: https://doi.org/https://doi.org/10.1016/j.iot.2019.100058.
25. Shaw M. Truth vs knowledge: The difference between what a component does and what we know it does. In: *Proceedings of the 8th International Workshop on Software Specification and Design*. pp. 181-186, IEEE Computer Society, Washington, DC, USA, 1996.
26. Singh L.K., Vinod G., Tripathi A.K. Impact of change in component reliabilities on system reliability estimation. *SIGSOFT Softw. Eng. Notes* 39(3), pp. 1–6, 2014.
27. Solat S. RDV: An alternative to proof-of-work and a real decentralized consensus for blockchain. In: *Proceedings of the 1st Workshop on Blockchain-enabled Networked Sensor Systems*. pp. 25–31. BlockSys'18, ACM, New York, NY, USA, 2018.
28. Tariq F., Colomo-Palacios R. Use of blockchain smart contracts in software engineering: A systematic mapping. In: Misra, S. Et al. (Eds.) *Computational Science and Its Applications – ICCSA 2019*. pp. 327–337. Springer International Publishing, Cham, 2019.
29. TELEKOM INNOVATION LABORATORIES Elements - An open source, sidechain-capable blockchain platform. [online] [accesat 03.01.2019]. Disponibil: https://bit.ly/2MT3SRE
30. Tosh D., Shetty S., Foytik P., Kamhoua C., Njilla L. CloudPOS: A proof-of-stake consensus design for blockchain integrated cloud. In: *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*. pp. 302–309, 2018.
31. Vale T., Crnkovic I., De Almeida E.S., Silveira Neto P.A.D.M., Cavalcanti Y.A.C., Meira S.R.D.L. Twenty-eight years of component-based software engineering. *J. Syst. Softw.* 111(C), pp. 128-148, 2016.
32. Voas J., Payne J. Dependability certification of software components. *Journal of Systems and Software* 52(2), pp. 165-172, 2000.
33. Wen Y., Lu F., Liu Y., Huang X. Attacks and countermeasures on blockchains: A survey from layering perspective. *Computer Networks* 191, 107978, 2021. [online] [accesat 12.03.2021]. Disponibil: https://doi.org/https://doi.org/10.1016/j.comnet.2021.107978
34. Wonjiga A.T., Peisert S., Rilling L., Morin C. Blockchain as a trusted component in cloud sla verification. In: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion. pp. 93-100. UCC '19 Companion, Association for Computing Machinery, New York, NY, USA, 2019. https://doi.org/10.1145/3368235.3368872.
35. Yang F., Zhou W., Wu Q., Long R., Xiong N.N., Zhou M. Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism. *IEEE Access* 7, pp. 118541-118555, 2019.
36. Zhou T., Li X., Zhao H. DLattice: A permission-less blockchain based on DPOSBA-DAG consensus for data tokenization. *IEEE Access* 7, pp. 39273 - 39287, 2019.