TAMPERE UNIVERSITY OF TECHNOLOGY

Blockchain Technology for Smartphones and Constrained IoT Devices

Citation
Zhidanov, K., Bezzateev, S., Afanasyeva, A., Sayfullin, M., Vanurin, S., Bardinova, Y., & Ometov, A. (2019). Blockchain Technology for Smartphones and Constrained IoT Devices: A Future Perspective and Implementation. In J. Becker, O. Pastor, E. Kornyshova, V. O. Korepanov, O. A. Tsukanova, J. B. Albornoz, D. Fedyanin, V. N. Burkov, D. M. Nazarov, D. Novikov, R. Uskenbaeva, ... A. V. Shchepkin (Eds.), *21st IEEE Conference on Business Informatics, CBI 2019* (pp. 20-27). [8808043] IEEE. https://doi.org/10.1109/CBI.2019.10092

Year
2019

Version
Peer reviewed version (post-print)

Link to publication
TUTCRIS Portal (http://www.tut.fi/tutcris)

Published in
21st IEEE Conference on Business Informatics, CBI 2019

DOI
10.1109/CBI.2019.10092

Take down policy
If you believe that this document breaches copyright, please contact cris.tau@tuni.fi, and we will remove access to the work immediately and investigate your claim.

Download date:22.01.2021

Blockchain Technology for Smartphones and Constrained IoT Devices: A Future Perspective and Implementation

Konstantin Zhidanov[‡], Sergey Bezzateev[⋆], Alexandra Afanasyeva[⋆],
Mikhail Sayfullin[‡], Sergey Vanurin[‡], Yulia Bardinova[⋄], Aleksandr Ometov[†]

[‡]Enecuum HK Limited, Hong Kong
[⋆]ITMO university, St. Petersburg, Russia
[⋄]Saint Petersburg State University of Telecommunications, St. Petersburg, Russia
[†]Tampere University, Tampere, Finland
Email: ceo@enecuum.com

***Abstract*—The blockchain technology is currently penetrating different sides of modern ICT community. Most of the devices involved in blockchain-related processes are specially designed targeting only the mining aspect. At the same time, the use of wearable and mobile devices may also become a part of blockchain operation, especially during the charging time. The paper considers the possibility of using a large number of constrained devices supporting the operation of the blockchain. The utilization of such devices is expected to improve the efficiency of the system and also to attract a more substantial number of users. Authors propose a novel consensus algorithm based on a combination of Proof-of-Work (PoW), Proof-of-Activity (PoA), and Proof-of-Stake (PoS). The paper first overviews the existing strategies and further describes the developed cryptographic primitives used to build a blockchain involving mobile devices. A brief numerical evaluation of the designed system is also provided in the paper.**

Keywords*—*blockchain, distributed systems, networks, applications, future perspective

I. Introduction and Motivation

The introduction of Bitcoin by Satoshi Nakamoto in 2008 had a significant impact on digital society [1]. As a first step, Bitcoin-like cryptocurrencies seemed an extremely innovative alternative financial paradigm. However, underpinning it lies a fascinating technological breakthrough: *blockchain technology*. The applications that could previously work only through trusted centralized entities can now operate without any constant connection to the authority while maintaining the same security and improving the overall system functionality [2]. This distinguishing feature has extended the implementation of blockchain beyond conventional cryptocurrency area [3].

The main idea behind blockchain itself lies in the concept of *trust* [4]. This idea is based on the fact that parties interacting in the system do not necessarily know or trust each other but still have an opportunity to transact securely. The use of blockchain eliminates the need for the involvement and continuous maintenance by the centralized 'trusted' authority, thus, enabling the network to operate in a distributed manner. True to its name, the records of transactions between nodes in a blockchain network are organized in a data structure known as "blocks". A series of blocks are arranged in a strictly increasing-time order by a linked-list like data style known as the chain of blocks, i.e., "blockchain". The blockchain is maintained as appending only local replicas of itself by the nodes participating in the replicated consensus process. Due to the blockchain property of immutability, it can be abstracted as a transactional system that enables a consensus to form within its participants [5]. The consensus holds unique probabilistic properties and can thus be leveraged as a fundamental building block for adaptive middleware that offers both deterministic and probabilistic consensus.

Most of the blockchain operation is based on a specially designed devices – miners, i.e., nodes attempting to solve the computational puzzles, in other words, to reach the Proof-of-Work (PoW) [6] for new block creating, and profit from the monetary compensation associated with it. Briefly speaking, a block contains 'nonces' that a miner must set in such a way that the hash of the entire block is smaller than a known target, which is typically a very small number. The difficulty of mining should be adjusted dynamically throughout the lifetime of the system [7].

The possibility to customize and style along with technological enhancements towards small-scale electronics and modern applications make handheld and wearable devices a strong contender in the Internet of Things (IoT) technological race [8]. Almost one billion wearable devices are expected to join the IoT family by 2021 [9]. Said fascinating development is a driving force behind the convergence of the physical and digital worlds that promises to create an unprecedented IoT market of 19 trillion USD over the next decade, and it is expected that a significant percentage of those devices will be smartphones.

There are currently about 2.71 billion smartphones in the world[1] and the average one can process 2 billion floating point operations per second (FLOPS)[2], thus, leaving us with

[1] See "Number of smartphone users worldwide from 2014 to 2020 (in billions)" by Statista, 2019: https://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/

[2] See "Processing power compared" by Experts Exchange, 2019: https://pages.experts-exchange.com/processing-power-compared

constantly underused 5 EFLOPS. This power can be used in the transaction publication and validation processes, smart contracts, or distributed storage [10]. Based on the above, almost any modern smartphone already has the power to be a part of Proof of Activity (PoA) [11] and execute related cryptographic primitives [12].

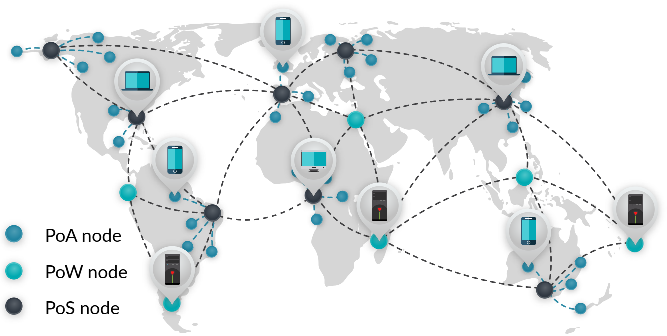

Fig. 1. Smartphones as part of blockchain ecosystem.

However, deploying blockchain applications to mobile devices acting as actual miners faces many critical challenges. It is mainly due to the mining process habits, i.e., solving the PoW that requires not only computing power but also energy from the interacting mobile devices. To address this challenge, the edge computing paradigm was introduced by the research community for cases of mobile blockchain networks [13]. However, it requires the use of more computational- and power-independent nodes to take block generation actions instead of actual smartphones. There are, however, a number of miner implementations of blockchain applications for smartphones but it has been shown that the income of a single mobile device acting as a miner in the blockchain network is nonprofitable[3].

Worth noting, the use of constrained devices is generally underestimated concerning the blockchain. The mining feature is ultimately not the most efficient utilization of this class due to the computational and power limitations, but the concept known as Proof-of-Stake (PoS) provided the first opportunity for such constrained devices utilization [14]. Here, PoS nodes do not act as miners to solve complex tasks. With PoS, stakeholders are used to confirming transactions and blocks based on their "stake" in the system and the use of the resource-constrained device, for this reason, is a natural step forward. The role of smartphones is to pay only the transaction fees of the network without involvement in actual mining. The probability of being selected to take part in the next block generation to the chain directly depends on the number of coins or tokens held by the relevant node.

The most interesting from smartphone perspective concept, proposed in [11], is indeed PoA. The authors envision a new protocol for a cryptocurrency constructed upon Bitcoin by combining the PoW component with a PoS type of system. PoA recommended itself as more secure against known practical attacks with relatively low utilization of both communications and storage resources. In PoA, mining is usually executed in a traditional PoW manner. However, the mined block does not contain transactions, i.e., the block is composed of the header and the rewards address. After the mining process, the system operation changes to PoS mode. Several stakeholders (smartphones, in our case) are randomly selected to sign (verify) the newly generated block. After everyone in the group signs the block, it is added to the blockchain. If some of the 'validators' have not participated in the validation process, the block is discarded, and the next PoW-based one is used, and the procedure is repeated. The reward is then split between active PoS validators and PoW miners.

The main contribution of this work is a protocol suite titled 'Trinity'. It allows utilizing an intelligent combination of PoA, PoS, and PoW aiming at the involvement of mobile devices for blockchain operation. The underlying concepts used in Trinity are as follows. The first one is *ID-based cryptography* initially proposed in [15] during the times when blockchain itself was brought to the research community's attention. After 20 years, the first realization of this strategy took place in work [16] by C. Cocks et al. They proposed a novel approach on obtaining the public key of the recipient for the signed message transmission employing Public Key Generator (PKG) and unique IDs of the participants. However, there is a number of challenges related to PKG utilization: (i) PKG can sign and decrypt all the messages; (ii) key revoking is not implemented; (iii) safe channel is required for the key dissemination; and (iv) encryption and decryption mechanisms are computationally different. Most of those could be mitigated by utilizing Shamir Secret Sharing [17] allowing for the secret key dissemination and reconstruction based on only a portion of previously distributed shares.

The rest of the paper is organized as follows. Section II provides the main primitives and constructs used to construct the Trinity. Next, Section III provides a detailed description of the developed cryptographic protocols. Section V provides a quantitative performance evaluation of the developed system. The last section concludes the paper and lists future research directions.

II. Main System Components

This section provides a brief overview of the main system components used during Trinity development. To start with, the block creation process begins when there are enough pending transactions to start assembling a block. Analyzing specified parameters of each transaction, miners determine its value for the system and add it into a corresponding block. Minimum-size blocks can be created to reach the minimum delay in speed per operation while possible, and as the load on the network increases, the block size grows. In circumstances where a user needs a block size larger than 4 MB, the system also supports combining any number of blocks (microblocks) into a macroblock, thereby allowing the storage of large volumes of data on the blockchain.

Bitcoin-NG protocol is selected to handle macroblocks [18] to reduce the latency between the creation of blocks so that each microblock inside a macroblock is created in real time and adds transactions to the blockchain immediately upon their arrival. So, there is no need to wait until an entire macroblock is completed, its hash is found, and it is synced between all

[3] See "Is Mobile Mining Profitable?", by COINCENTRAL, 2018: https://coincentral.com/is-mobile-mining-profitable/

nodes on the network – small microblocks can be generated concurrently inside it. The main reason behind the utilization of this protocol lies in its possibility to increase the mining speed in the system, i.e., to increase the number of blocks generated by the system within selected time frame. The fundamental limit here is the distribution time of the newly generated block between all the nodes in the system. In case the generation time is smaller, the probability of forking in two distant sections of the network may arise tremendously. Direct Acyclic Graph (DAG) [19] allows the addition of new blocks in different network segments handling the forking.

The goal of DAG is to deterministically rearrange the k-blocks for the ledger recalculation based on the following set of requirements:

- Graph construction and graph walk procedures are developed minding the *consensus* between the nodes, i.e., there is a need for defining the minimal number of nodes to guarantee the validity of current system state at any time of execution;
- New k-block is validated (added to consensus) during a specific time frame;
- New k-block should be inserted in the chain according to its publishing time;
- Addition of a new k-block should not require the traversal of the entire graph;
- Long-time forks should be avoided.

A. Proposed utilization of DAG

First, the graph walk procedure is defined, starting with inverting the DAG. Next, the Queue-based topological order algorithm is applied to the graph as by iterative removing of the nodes and storing the logs of this process, see [19]. We assume that there exists the deterministic algorithm allowing to calculate the difficulty for each k-block during the graph traversal. Thus, every new k-block is considered valid if its' hash is equal to its' difficulty. We also assume the deterministic algorithm allowing to calculate the value $branch\_max$ during the graph traversal based on the k-block $number$, *brunch* ($0 < branch < branch_{max}$). Each k-block s has two links to previous and next k-blocks t_1 and t_2 such that $t_1.branch \ == s.branch$ and $t_1.branch \ != \ s.branch$ despite the case when $branch_{max} = 1$.

New k-block generation procedure is described as follows. First k-block has $branch = 0, number = 0$. It is valid if:

1) $\{number, branch\}$ pair is unique;
2) k-block has links to t_1 and t_2, $t_1.branch \ == s.branch$, $t_1.branch \ != \ s.branch$, $s.number > t_1.number$. In case there are more than one s, the one with higher $t_2.number$ will be accepted;
3) k-block's hash is equal to $difficulty$.

B. Ledger operation

The following parameters are considered during the ledger calculation: k-block mining; a reward for microblock publishing; and transaction fee. The rewards are dynamic and based on the blockchain operation history. The transactions inside the microblock are stored in a sorted array. Therefore, all k-blocks, microblocks, and transactions could also be arranged for any DAG size. As a result, the entire history of events could be linearly retrieved, thus allowing to calculate the states of the account balance.

At the beginning of the execution, the ledger is empty. During the block rewarding process, the balance of the existing account will be changed, or a new record will be found. The states of nodes are updated during the transactions accordingly. The transaction is treated as invalid if there is no information about the account in the ledger or it has not enough coins in the wallet. Invalid transactions are discarded.

C. Rewarding and Difficulty Estimation Policies

The estimation of the reward is based on the deterministic algorithm for each system state based on history and the current block. The estimation of rewards depends on the emission curve and current emission distribution. Initially, the distributions are as follows: PoW – 10%; PoS – 25%; and PoA – 65% of the emission. The emission distribution balance is a dynamic system property and could be used as a tool to mitigate malicious activity between different nodes based on a specifically selected emission curve. Generally, the values of rewards are estimated in such a way that it is inexpedient to run PoA emulators on the hardware suitable for PoW or PoS.

Authors have designed a reward and difficulty assignment system, Neuro, based on the recurrent neural network. Neuro uses historical blockchain data to predict the required rewards and difficulties for each new cycle. As soon as a cycle is completed, the statistics of that cycle are used to improve the network's next predictions. To make these predictions, we make use of a variation on a type of neural network that has a selective long-term memory: a recurrent neural network. For the non-recurrent neural network, each forward cycle starts with a clean state, and neurons have values that originate only from weighed connections to neurons in the previous layers (or inputs). A recurrent neural network is a network where the result of a neuron activation, the state, affects the next forward cycle of the network.

D. Trinity consensus

The following subsection describes the interaction between nodes during the blockchain construction.

The system is based on three types of users: (i) solver (PoW); (ii) holder (PoS); and (iii) publisher (PoA). None of those could take the role of another, which is achieved by cryptographic and technical methods. The following describes how the blockchain operation is divided by those nodes. Note, none of the types can form the blockchain independently.

PoW solver is responsible for the generation of new k-blocks. The main requirements for this type of nodes are (i) reliable access to the Internet; (ii) storage (required to store the blockchain structure); and (iii) computational power for hashing. The solver is recursively calculating nonces for new k-block generation according to the set of predefined rules – difficulty, batch number, hash links validity. Each k-block is distributed through the network in a broadcast way after its generation. Each node is checking its validity based on locally stored data and add it to local blockchain storage if valid.

Widely known Nakamoto protocol [1] is used for the blockchain construction. The solver's main aim is to generate the block and obtain the resulting award for the computational expenses. The k-block contains its solver's public key. The selection of the hashing function does not affect the overall system operation directly. Presently, we utilize SHA-256 for performance evaluation, but this choice is temporal since modern ASICs can easily calculate it.

The PoS holder is a node holding a significant amount of coins. The node can prove the eligibility to become a holder. Key SK_{HK} is a shared key distributed between a set of holders based on the Lagrange interpolation formula [20]. The corresponding PK_{HK} is known to any node. The *resident* node is responsible for SK_{HK} generation, and a group of holders forms a Private Key Generator (PKG).

Holders are systematically executing the protocol described in subsection III-A to verify who has the right to distribute the publication keys during this system operation state. The corresponding time interval is set to 100 k-blocks in the simulation environment. The Leading PoS (LPoS) selection result is then stored as statistic blocks and may be verified by any node.

Next, all PoSs are generating the publication secret key for LPoS after the k-block retrieval. The publication public key is calculated based on the k-block ID (hash sum) and LPoS ID. The secret key and the corresponding shares are calculated based on the protocol described in subsection III-B. The intermediate execution results are stored in the static block and could be verified later on. The holder gets a reward for participation in the voting and PKG-related procedures.

PoA publishers are involved in the microblock publishing process. A coalition of grouped PoA publishers should generate each microblock. In order to retrieve a new k-block, a group of PoA generates the microblock payload (array of transactions) and forwards it to the LPoS. After the LPoS verifies each microblock, it puts the signature by using the current publication secret key. Therefore, the microblock data becomes validated by PoA node and LPoS node in the system, and their participation may be verified later. The PoA rewards are based on participation in the verification procedure.

The resident is one of the PoS holders of the system being controlled by the blockchain itself. The resident node is active only during some period of the initial system operation, and its function may be automatically distributed between the other PoSs in the network.

The resident's functions are: (i) to store SK_{HK}; (ii) to distribute it to other PoSs; and (iii) to estimate the Lagrange polynomial properties. After the resident is stopped, the key shares will be distributed to PoS according to protocols described in subsection III-G and III-H. The Lagrange polynomial characteristics would not be possible after the resident leaves the system and, thus, they should be adjusted after the initial period of the system operation.

III. Developed cryptographic protocols

This section provides a brief overview of the developed protocols. More details on the implementation could be found in [21], more recent versions of protocols (if any changes) would also be available via the link.

Each k-block has its unique ID_k estimated according to correct execution of function $f(*)$ as

$$ID_k = f(block_k), \quad (1)$$

where f is the desired hashing function.

A. Protocol of the "leading" PoS miner selection during the session (voting)

The main requirements of the protocol are resistance against the repetitive selection of the same miner during a series of sessions, i.e., improved randomization; and protocol should be executed either by a group of PoS miners or the entire available set but the selection rule is different for each execution.

To exclude the possibility of restarting elections by disgruntled PoS miners, the result should be pseudo-random but rigidly deterministic current k-block and list of voters. A series of assumptions are thus introduced:

1) All PoS miners in the state of the Ledger stored by them can compile a list of all PoSs. In this case, all sets of identifiers will be obtained identically and ordered lexicographically.
2) In the course of the routing procedures, each PoS miner compiles a list of currently active PoS. At the same time, the lists of participants differ by no more than 10%. The list is stored as a binary vector: $V_{PoS} = (0, 1, 1, 1, 0, \dots, 1)$, where the number of positions coincides with the size of the list from item 1, "0" means that the participant with the given identifier is inactive, and "1" that it is active.

With this list and its associated vector, each node can vote.

Stage A: After the list is constructed, each participant (PoS miner) calculates the hashing function

$$r = \frac{Hash(ID_k|PoS_1|\dots PoS_N)}{Hash_{max}} \in (0, 1), \quad (2)$$

where $Hash_{max} = maxHash(ID_k|PoS_1|\dots|PoS_N)$.

Therefore, the voting is further based on r and on comparing it to a newly generated discrete random variable in the same bound. Thus, each PoS_i receives a probabilistic value based on his public rating. The sum of all PoS probabilities should be equal to 1. After that, the probabilities are logically interpreted into intervals on the section from 0 to 1, and the tagged PoS node is selected if r is located in its interval.

Stage B: After the tagged PoS was selected (LPoS status), the voter calculates corresponding publication public key and transmits the secret key share to selected LPoS. After one PoS receives at least k of shares (basically, those have the same list on their side), the secret key is generated as described in protocol III-B.

Stage C: LPoS forms an entry to the static block after the session key is received. The entry is formed from the k-block number and voting list signed with the session key. Thus, it becomes possible to validate LPoS rights and distribute rewards.

B. Leading PoS miner key generation

The main requirements set to the protocol are: keys could only be used once; keys should be distributed securely; any user could not generate keys; and keys do not contain any information related to PoS miner secret keys.

The protocol execution could be done in case the leading miner is selected by $LPoS = PoS_i$ according to the described protocol. Each PoS has its pair of keys PK_{PoS_i}, SK_{PoS_i} directly related to his wallet.

Next, the session key PK_{LPoS} is generated for leading PoS. It will be further utilized for the microblocks signature and, thus, would be split into shares and distributed between PoAs. PK_{LPoS} is defined by k block present in current session and ID_{LPoS}. ID_{LPoS} is selected as PK_{LPoS} or a function of this key. PK_{LPoS} and SK_{LPoS} would be thus selected as

$$PK_{LPoS} = f(block_k||ID_{LPoS}), \tag{3}$$

$$SK_{LPoS} = newSK_{LPoS}. \tag{4}$$

SK_{LPoS} is generated by PoS miners according to the distributed ID-based cryptographic PKG method [22] by k of n schema. Which considers the collision resolution for cases when more that one leader is selected.

Algorithm 1 Initialization of ID-based schema with distributed PKG

```
Define groups:
    Define $G1$ as a cyclic group of order $q$ (group of points on elliptic curve);
    Define multiplicative group $G2$;
Define functions:
    $H1 : (0,1)* \to G1$;
    $H2 : G2 \to (0,1)*$;
    $H3 : (0,1)* \to Zq$;
    $e : G1 \times G1 \to G2$ (bilinear mapping);
Define Master Secret Key (MSK) as $s \in Zq$;
Define $P$: generator of $G1$;
Define Public Master Public Key (MPK) as $s * P$.
```

Algorithm 2 PKG (k, n) Master Secret Key splitting

```
Generate random polynomial in residue field $q$: $deg(\phi(x)) = k - 1$;
Each participant (PoS miner) receives its key share of Master Secret Key $ss_i = \phi(ID_i) mod\ q$.
```

Algorithm 3 Session key SK_{LPoS} generation for $LPoS$

```
Each of $k$ participants calculates $PK_{LPoS} = f(block_k||ID_{LPoS})$ according to equation (3).
Transmits its $ss_i \cdot PK_{LPoS}$ and $ID_i$ to LPoS.
```

Algorithm 4 Secret key recovery

```
LPoS is calculating $SL_{LPoS}$ based on the received from algorithm III-B data as
$SK_{LPoS} = \sum_{i=1}^{k} \lambda(ID_i, 0) ss_i \ PK_{LPoS}$, where $\lambda(ID_i, 0)$ is a Lagrange coefficient generated per coalition for each user $ID_i$ and 0.
```

C. Protocol of the PoA applicability for microblock generation procedure

The coalition of PoA miners is selected after new k-block is published. It is selected based on constant N_{PoA} per node and the corresponding ID such that $Hash(PoA_{ID}) = Hash(k - block||i), i = 1, \ldots, N_{PoA}$. Therefore, each node has an opportunity to verify if his ID is in the group fast, while brute-force attack on the ID is a computationally complex task.

D. Generation of microblock by PoA for current k-block

The main requirements for the protocol are simultaneous and independent execution of the coalition members; data exchange minimization; in-block additional data minimization; and confirmation of the participation in the verification process.

Each PoA miner verifies if it is applicable for new microblock generation III-C after new k-block is published. In case applicable, it forms a new microblock M based on the selected transaction with a predefined size. After M is formed, PoA adds the following data to it: PoA_{ID}, k-block number. Next, it is signed by its' SK_{PoA} and immediately published.

E. LPoS microblock assurance protocol

After protocol III-A is executed and the new session key is generated with protocol III-B, LPoS starts to assure the microblocks. *Stage A*: After PoAs have published the corresponding microblocks, LPoS is collecting those from the network. LPoS is verifying the k-block number and verifies if PoAs are in the coalition of this block.

Stage B: LPoS verifies the validity of transactions in the microblock based on the ledger. *Stage C*: In case the verification succeeds, each microblock is signed with SK_{LPoS} from protocol III-B according to Algorithm 5.

Algorithm 5 Microblock signature protocol

```
LPoS generates $r$ from $Zq$;
Calculates $R = rP$ and
    $$\begin{aligned} S = SK_{LPoS} + rf(ID_{LPoS}, M) = \\ = sQ + rf(ID_{LPoS}, M), \end{aligned} \tag{5}$$
    where $M$ is the entire microblock;
Adds $(R, S)$ to the microblock.
```

Next, PoW miner is in standby mode until the required number of transactions is collected, and generates new k-block for all the obtained microblocks.

F. Cryptographic microblock verification protocol

The main goal of the protocol is to check the created microblock at any time, and the requirements are: it should be executable at any node; it should be based only on publicly available information. Two signatures verify each microblock: the first one is the signature of PoA miner generating microblock, and the second one is the signature of LPoS miner. The verification procedure is made according to algorithm III-F.

Algorithm 6 Cryptographic microblock verification protocol

```
At first the verifiers checks the k-block number, and then that the PoA-miner is a member of the group of publishers for this session and his signature.
Then it calculates $PK_{LPoS} = f(block_k||ID_{LPoS})$ according to equation 3.
Then the verify node check the signature of the microblock $R$ and $S$ by
    $e(P, S) =$                                                  (6)
    $= e(MPK, PK_{LPoS} = Q) \cdot e(R, f(ID_{LPoS}, M)),$
    where $P$ is a generator of G1, $MPK$ is a Master Public Key and $M$ is a microblock.
```

The microblock is assumed as verified if both signatures are verified successfully.

G. Distributed PKG secret update

This phase is executed either whenever the set of PoS miners changes, or during the ledger recalculation when any of the PoS nodes loses the PoS status. The resident node distributes new key shares. It is also responsible for the (k, n) relations during the initial system operation stage. After the system operation is stable, its role is distributed between PoSs.

H. Distributed PKG new secret share transmission protocol

When a new node arrives (reaching the border or if some other condition is met), the new node requests its share of PKG master secret key. If it has the right, the resident node responds. Keys to new participants are built and given out by the resident node, while the system developer acts in his role. When the parameters are settled, the resident role can be dissolved in PoS miners. For a new participant, his polynomial point is calculated as

$$ss_i = \phi(ID_{new}) mod\ q, \tag{7}$$

where q is the order of the group of points G1.

By efficient integration of the previously developed protocols, it would become possible to involve a high number of recourse-constrained devices in the blockchain operation.

IV. Simulation environment description

For the performance evaluation campaign, we selected project *p2psim* mainly because it has native support for Chord emulation, which is essentially our graph topology. Moreover, it has a set of real packet propagation measurements between thousands of nodes, collected in *kingdata* package. Next, we employ a big number of simultaneously opened TCP socket connections between the nodes aiming at decreasing the delay. We detail the system model in the following.

In this paper, we provide an example of the system operation evaluation from communication (signaling) perspective. The main focus is given to 'tagged' LPoS and the packet transmission time between related nodes and the corresponding packet processing and storing (interaction with the database) metrics. The PoS nodes (acting as PoW) are generating new blocks while PoA nodes are adding those to the blockchain.

Generally, packet exchange is present between (i) Chord nodes, i.e., PoSs based on TCP; or (ii) PoA to PoS nodes. The communication in the second scenario is organized directly from PoA to first PoS node and further through the Chord (executing the Chord routing). PoA nodes could be classified as "data" stored in Chord. The details of the Chord consistency are omitted in this document but could be checked in [23]. The broadcast procedure is balanced according to [24].

The message sizes utilized in this campaign are microblock – 100kb (approximately 650 transactions); others – 144 bytes (1 transaction). Table I provides an overview of the main message types and relative load. Therefore, additional Chord → Chord messages can provide a significant load on the LPoS. Precisely, this may happen while receiving replies from PoA → LPoS.

Next, we will focus on the packet propagation time faced by our system. The use of TCP for our system generally increases the Round Trip Time (RTT)/delay in a trade-off to reliability. The approximations used in this campaign are based on the public data[4,5]. Note, potential higher delays faced by the cellular network users are not expected to affect our system operation. The locations of PoW and PoS nodes are hard to predict while PoAs are expected to be mobile nodes. Thus, the system analysis should also consider the delays between the main operator's gateways. The detailed data on the measurements could be found in [21].

V. Selected numerical results

In the following, n is the number of network nodes, k is an average number of PoA nodes per PoS, lat_a is the average delay between PoS → PoA, lat_{ch} in the average delay between Chord's nodes, bw_{ch} in the node processing speed in Mbps, $disk_{speed}$ is LPoSs' average database interaction speed, kbl_{size} is the k-block size, mbl_{size} is the microblock size, $msig_{size}$ is signed microblock from PoA.

We aim at finding the limitations of the LPoS (regarding maximization of k) varying the number of PoAs in terms of operational delay and, based on the above, we have quantitatively analyze the number of signaling messages required for different messages dissemination. The simplified results are shown in Table II.

Note, that real life timings may be less optimistic due to our simplification and averaging of lat_{ch}. The results of a more realistic system operation estimates are also presented

[4] See "Global Ping Statistics: Ping times between WonderNetwork servers", 2018: https://wondernetwork.com/pings

[5] See "Ookla Speedtest, and Speedtest Intelligence", 2018: http://www.speedtest.net

TABLE I. APPROXIMATE NETWORK LOAD BY DIFFERENT MESSAGE TYPE

Type	Source	Destination	Other Chord nodes	Other Chord nodes average #	Example: n = 300
A → S	1	1	-	-	-
S → A	1	1	-	-	-
Chord → Chord	1	1	$< log(n)$	$((log((n))/2-2))/((n-2))$	0.0084/0.0084
Broadcast Chord → Chord	2	1	Each node receives one and transmits either zero or two	1/1	1/1
Broadcast Chord → PoA	2	1	PoW like in Chord → Chord and each PoS transmits the related to his PoA messages	$1/\ldots$	$1/\ldots$

TABLE II. DESCRIPTION AND THE NUMBER OF MESSAGES REQUIRED FOR BLOCK'S DISSEMINATION

Time	LPoS	Chord's nodes	PoAs	Optimistic action	Pessimistic action
0				PoW's k-block transmission	-
$4 * lat_{ch}$	k-block received	k-block received by 50%	Started to receive k-block	$k * kbl_{size}/bw_{ch}$ required to deliver k-block to PoAs	$k * kbl_{size}/bw_{ch}$ required to deliver k-block to PoAs. For example, $0.05 * k * lat_{ch}$.
$5 * lat_{ch}$	$shadow_{request}$ transmitted				
$13 * lat_{ch}$	Received $shadow_{response}$	k-block received	k-block received by the majority	LPoS is ready to send $leader_{beacon}$	If $k \leq 400$ – k-blocks are delivered to PoAs.
$23 * lat_{ch}$		The remaining ones are transmitting $leader_{beacon}$ to PoAs	Receiving $leader_{beacon}$	LPoS's disk utilization increases. $mblock_{sign}$ arrival begins	**If $k \leq 600$ – k-blocks are already delivered to PoA**
$33 * lat_{ch}$	Majority received			LPoS disk load is still present	$k * n * msig_{size}/disk_{speed}$ equals $k * n/180000$ seconds
$45 * lat_{ch}$	Last $mblock_{sign}$ received			LPoS starts to broadcast mblock via Chord.	Either the disk utilization is finished, or k-block distribution is finished.
$54 * lat_{ch}$		mblock received		The procedure is over. Total time is around 8 seconds.	

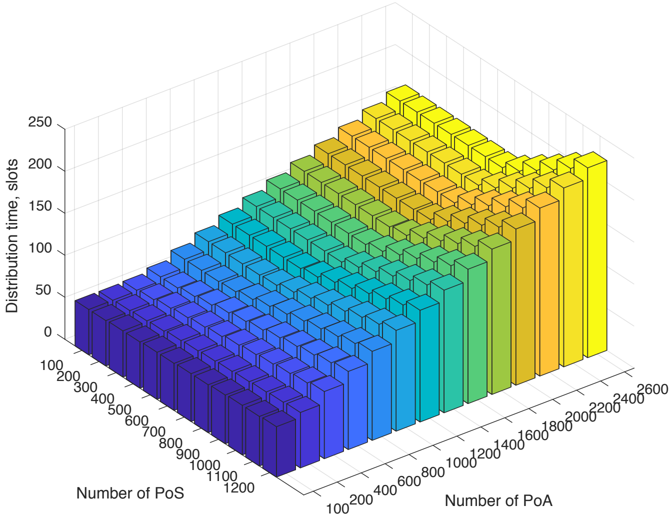

Fig. 2. System operation time varying number of PoSs and PoAs

in Table II in the last column. Based on the results, the pessimistic estimation of the time required for the new k-block creation is $max(53 + k * n/18000, 30 + 0.05 * k)$. Next, we provide a graph with the effects of k and n relation, see Fig. 2. Note, $lat_{ch} = 0.150$ seconds, so that 30 seconds equals 200 timeslots. It could be concluded that the variation of the PoS nodes number does not have a significant impact on the communications delay due to highly predictable packet propagation time through the Chord nodes. On the other hand, increasing the number of PoA nodes has a more significant impact due to the need to communicate through the more complex network infrastructure.

VI. Conclusion

The penetration of blockchain technology in our daily lives could not be stopped, therefore, we have proposed a 'Trinity' concept coupling together Proof-of-Work, Proof-of-Stake, and Proof-of-Activity blockchain strategies motivated to involve mobile devices in the new block generation process since the computational resources of said devices are underused today. Currently, we are in the final phase of the developed primitives implementation in our custom simulation environment, and some quantitative numerical results are already presented in the paper. At the moment of the paper submission, authors already implemented the testnet involving more than 500 accounts in 22 countries[6].

[6] See "Development Report May 2019 #2", by ENQ Enecuum Blockchain, 2019: https://medium.com/@ENQBlockchain/development-report-may-2019-1-2b7f1f92322e

As for the future work, we aim at evaluating the effects of network parameters (throughput, latency, propagation delays, etc.) of the system operation, detailed study on a variety of security and privacy threads of the developed system, and actual benefits on the developed system utilization for the end users.

Acknowledgment

Work of the last author is supported by Nokia Foundation under a personal grant.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[2] A. Hari and T. Lakshman, "The Internet Blockchain: A Distributed, Tamper-resistant Transaction Framework for the Internet," in *Proc. of 15th ACM Workshop on Hot Topics in Networks*, pp. 204–210, ACM, 2016.

[3] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[4] F. Hawlitschek, B. Notheisen, and T. Teubner, "The Limits of Trust-free Systems: A Literature Review on Blockchain Technology and Trust in the Sharing Economy," *Electronic commerce research and applications*, vol. 29, pp. 50–63, 2018.

[5] N. Chalaemwongwan and W. Kurutach, "State of the Art and Challenges Facing Consensus Protocols on Blockchain," in *Proc. of International Conference on Information Networking (ICOIN)*, pp. 957–962, IEEE, 2018.

[6] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," in *Proc. of ACM SIGSAC Conference on Computer and Communications Security*, pp. 3–16, ACM, 2016.

[7] M. Vukolić, "The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication," in *Proc. of International Workshop on Open Problems in Network Security*, pp. 112–125, Springer, 2015.

[8] A. Ometov, V. Petrov, S. Bezzateev, S. Andreev, Y. Koucheryavy, and M. Gerla, "Challenges of Multi-Factor Authentication for Securing Advanced IoT Applications," *IEEE Network*, vol. 33, pp. 82–88, March 2019.

[9] Cisco, "Global Mobile Data Traffic Forecast 2017–2022." White Paper, 2019.

[10] D. Frey, M. X. Makkes, P.-L. Roman, F. Taïani, and S. Voulgaris, "Bringing Secure Bitcoin Transactions to Your Smartphone," in *Proc. of 15th International Workshop on Adaptive and Reflective Middleware*, p. 3, ACM, 2016.

[11] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, "Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake [extended abstract]," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 3, pp. 34–37, 2014.

[12] A. Ometov, P. Masek, L. Malina, R. Florea, J. Hosek, S. Andreev, J. Hajny, J. Niutanen, and Y. Koucheryavy, "Feasibility Characterization of Cryptographic Primitives for Constrained (Wearable) IoT devices," in *Proc. of PerCom Workshops*, pp. 1–6, 2016.

[13] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal Pricing-based Edge Computing Resource Management in Mobile Blockchain," in *Proc. of IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2018.

[14] S. King and S. Nadal, "Ppcoin: Peer-to-Peer Crypto-currency with Proof-of-Stake," *self-published paper, August*, vol. 19, 2012.

[15] A. Shamir, "Identity-based Cryptosystems and Signature Schemes," in *Proc. of Workshop on the Theory and Application of Cryptographic Techniques*, pp. 47–53, Springer, 1984.

[16] C. Cocks, "An Identity Based Encryption Scheme Based on Quadratic Residues," in *Proc. of IMA International Conference on Cryptography and Coding*, pp. 360–363, Springer, 2001.

[17] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[18] E. Heilman and T. Dryja, "IOTA Vulnerability Report: Cryptanalysis of the Curl Hash Function Enabling Practical Signature Forgery Attacks on the IOTA Cryptocurrency [OL]." Technical Report, September 2017.

[19] A. B. Kahn, "Topological sorting of large networks," *Communications of the ACM*, vol. 5, no. 11, pp. 558–562, 1962.

[20] A. Ometov, K. Zhidanov, S. Bezzateev, R. Florea, S. Andreev, and Y. Koucheryavy, "Securing Network-Assisted Direct Communication: The Case of Unreliable Cellular Connectivity," in *Proc. of IEEE 14th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, IEEE, 2015.

[21] ENECUUM HK LIMITED, "Dynamic Mobile Blockchain with Enecuum: A Synergy of Proof-of-Work, Proof-of-Activity, and Proof-of-Stake." White Paper [Online] https://new.enecuum.com/files/tp_en.pdf, 2019.

[22] C. Gentry and A. Silverberg, "Hierarchical ID-Based Cryptography," *Proc. of Advances in Cryptology (ASIACRYPT)*, vol. 2501, 2002.

[23] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001.

[24] K. Huang and D. Zhang, "Dht-based lightweight broadcast algorithms in large-scale computing infrastructures," *Future Generation Computer Systems*, vol. 26, no. 3, pp. 291–303, 2010.