Dynamic Mobile Blockchain with Enecuum: A Synergy of Proof-of-Work, Proof-of-Activity, and Proof-of-Stake

Enecuum HK Limited

Rm 1202, West Exchange Tower, 322 Des Voeux Road Central, Hong Kong

Contact us: hello@enecuum.com

May 23, 2019

Currency is to the economy what language is to speech. Its sophistication and usefulness evolve in direct proportion to its user-base, to the volume of "transactions" keeping it alive and to its Darwinistic ability to adapt, proliferate and thrive. Most traditional currencies, as with languages, developed naturally, their rules and features crystalizing through time and adoption. So too it is with digital currencies, as the vision illuminated by Satoshi Nakamoto in the 2008 Bitcoin whitepaper, of a "purely peer-to-peer version of electronic cash", gains momentum, and the technology which underpins it evolves into a truly revolutionary gravitational force.

The inbuilt architectural limitations of existing blockchain protocols typically demand a compromise between security, speed, and scalability. To harness the true real-world potential of this technology, therefore, requires an evolution of its design. Enecuum seeks to offer this solution by introducing an innovative platform unconstrained by existing protocol design limitations. Accordingly, the technical descriptions offered in this whitepaper all support the same core idea: speed, security, and scalability – the blockchain of tomorrow.

Contents

1 Introduction

1.1 Background

The introduction of Bitcoin by Satoshi Nakamoto in 2008 had a significant impact on modern 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. Thanks to blockchain technology, applications that could previously work only through trusted central entities can now operate without a centralized authority while maintaining the same security and improving functionality. This distinguishing feature has extended the implementation of blockchain beyond the conventional cryptocurrency area [2].

The main idea behind blockchain itself lies in the concept of trust. 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 of a 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. Because of blockchain immutability, it can be abstracted as a transactional system that enables a consensus to form within its participants. This consensus holds unique probabilistic properties and can thus be leveraged as a fundamental building block for adaptive middlewares that offer both deterministic and probabilistic consensus [3].

At the same time, the subsequent emergence of smart contracts took its place. The utilization of smart contracts facilitates credible automated transactions on pre-determined conditions, significantly expanding the application potential of this technology. It is believed that blockchain technology is capable of revolutionizing many areas of financial and economic activity, such as trade, financial markets, voting, and even logistics.

1.2 Use of mobile devices for Blockchain

Most of the blockchain operation is based on specially designed devices – miners, i.e., nodes attempting to add a block to the chain. They try to solve Proof-of-Work (PoW) computational puzzles to create new blocks, and profit from the monetary compensation associated with it. Being short, 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 [4].

The possibility to customize and style them along with technological enhancements towards small-scale electronics and modern applications make wearables a strong contender in the IoT technological race. Almost one billion wearable devices are expected to join the IoT family by 2021 [5]. This fascinating development is a driving force behind the convergence of the physical and digital worlds that promises to create an unprecedented Internet of Things (IoT) market of 19 trillion USD over the next decade, and it is expected that a significant percentage of those devices will be smartphones.

While the total computational power of smartphones in 2017 was said to be 1250 petaflops, only 10% of that power was used. It would be a waste to pass on this opportunity. This power can be used in the transaction publication and validation processes, decentralized storage and help to enable smart contracts. Which, in turn, could also be used while constructing a blockchain. As the estimated minimum requirements for a mobile device to run a Proof of Activity (PoA) [6] node are: Android 5.0, 1GHz CPU and 1Gb RAM.

However, deploying blockchain applications to mobile networks acting as actual miners faces many critical challenges. This is mainly due to the mining process habits, i.e., solving the PoW puzzle, which requires high computing power and energy from the interacting mobile devices. To address this challenge, the edge computing paradigm was introduced by the research community for cases of combined mobile blockchain networks [7]. This, however, requires the use of more computational and power-independent nodes to take actions instead of actual smartphones.

There are, however, a number of miner implementations of blockchain applications for smartphones. According to studies, it has been shown that the income from a single device acting as a miner in the blockchain network is nonprofitable.

The use of smart devices is generally underestimated concerning the blockchain. The mining feature is ultimately not the most efficient utilization due to the computational and power limitations, but the concept known as Proof-of-Stake (PoS) provides an excellent opportunity for such constrained devices utilization [8]. Here, PoS does not utilize miners to solve complex tasks. With PoS, stakeholders are used to confirm 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.

In PoS, 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 generate the next block in the chain directly depends on the number of coins or tokens held by the relevant smartphone.

The next hybrid concept proposed in [6]is so-called PoA. The authors proposed 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 a 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. A number of stakeholders 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.

Enecuum aims to involve smartphones to execute PoS mining as part of the PoA paradigm.

This is why 65% of the ENQ emission is intended for PoA mining, while 10% is for PoW and 25% reserved for PoS mining.

1.3 From the beginning to Enecuum – technical aspects

Overall, Enecuum itself is not a protocol developed from scratch. Namely, it utilizes a number of solutions known to the community for many years and the intelligent combination of those made it possible to develop Enecuum.

Starting from the basics, there is a must to differentiate between the term *consensus* and *blockchain*. First one represents the case when there are two data structures on two different nodes, and there is a must to find the actual one. The roots of this problem could be found in work 'The Byzantine generals problem' by L. Lamport et al. from early 80th [9]. Here, the required number of nodes needed for the verification is proven analytically. If there are n nodes and k out of those are malicious ones than the rest should reach the consensus in such an uncertain situation.

As a next step, A. Back discussed a solution to enable a denial of service counter-measure based on blockchain technology as a way to resist against spam [10] followed by the Nakamoto' PoW concept [1]. Worth noting that Nakamoto had a significant assumption in his work. In particular, he assumed that one node of the network could only have one processing unit eligible for one participation in the new block generation race. This was, however, mitigated with the introduction of numerous blockchain farms.

Later on, the scalability issues of Nakamoto's solution became present. In particular, the system throughput is directly dependent a number of factors: (i) the block generation frequency and block size; (ii) block generation interval should be larger than the block propagation to the majority of the participating nodes interval (in order to resist to the forking problem); (iii) one way to increase the throughput is to increase the block size; (iv) latency between the nodes is a significant metric profoundly affecting the system throughput.

Many blockchain researchers claim that the blocksize continuous increase is not an option for the system throughput management[1]:

1. It is necessary to wait for sufficient consensus in case of a hard fork;
2. There is always a risk of catastrophic consensus failure;
3. An emergency hard fork that can achieve consensus can be deployed on a short period if needed;
4. Orphan rate amplification, more reorgs and double-spends due to slower propagation speeds;
5. "Congestion" concerns can be solved with mempool improvements including transaction eviction;

[1]See 'Block size limit controversy': `https://en.bitcoin.it/wiki/Block_size_limit_controversy`

6. No amount of max block size would support all the world's future transactions on the main blockchain;

7. Fast block propagation is either not viable, or creates centralized controls.

Further on, work [11] provided an extension to Nakamoto's work enabling for the differentiating the consensus algorithm (mainly, PoW) and the data structure management. By adding macroblocks and microblocks to the Nakamoto's consensus. Each miner is attempting to generate the next k-block instead of direct publishing of the block. However, the winner of the race obtains an opportunity to submit the microblock to the blockchain, i.e., the block containing the actual transaction. Note, microblock is generally smaller than the macroblock, thus it brought new opportunities to the overall system operation such as, for example, lower latencies and more flexible rewarding system for both microblock and macroblock generation procedures.

Next, work [6] presented the novel PoA concept in 2014. Here, each miner still aims at generating the block *header*, instead of the entire block, which is still based on the Nakamoto's consensus. The header is further broadcasted through the network where each node is receiving a list of active *stake holders* (nodes that already have some stake on their disposal) based on a predetermined function [8]. Stakeholders are the only nodes with a right to publish microblocks. However, no present systems are utilizing this concept. So, Enecuum decided to base its operation on PoA concept.

The baseline of Enecuum relies on a number of concepts. The first one is *ID-based cryptography* firstly proposed by A. Shamir [12] very close to 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 [13] by C. Cocks et al. proposes 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 [14] allowing for the secret key dissemination and reconstruction based on only a portion of previously distributed shares. Namely, there are strategiesallowing to sign a message utilizing the key share and validate it based on k collected signatures of this type.

Again, Enecuum is based on all of the enablers listed previously, and our central concept is drawn in Section 3.13.

1.4 White paper structure

This paper is organized as follows. In Section 2 provides an overview of related work from both academia and market perspectives. Section 3 describes the developed protocols and features of Enecuum. Next, Section 5 provides the conclusions and future work. Main literature sources are given in the next section. Additional information is given in the last section.

2 Related work and concept

2.1 Motivating techniques

Enecuum is designed as a decentralized blockchain platform of the next generation with unique features that have the potential to help with implementing a large number of secure and scalable blockchain services and decentralized applications.

One of Enecuum's critical advantages over other platforms is the Directed Acylic Graph (DAG) protocol, which is a data model for storing and writing transactions, with flexible settings offering new opportunities for the practical application of blockchain technology. DAG supports the creation of separate branches in which rules can be tailored to solve numerous potential business problems including the ability to handle a large number of transactions cheaply and quickly. Furthermore, this solution allows integrating smart contract technology that is successful in solving the scalability problem.

Linear logic allows for reliable automatic certification of smart contracts before their publication to the system, which we believe significantly reduces potential vulnerabilities, misuse, freezes, deadlocks, and other undesirable outcomes in the system.

2.2 Benefits of Enecuum

Another advantage of Enecuum is that it is a highly adaptive system. Users can take part in its development and vote for other participants' proposals for improving system functionality. There are two ways to factor in changes of the system parameters:

- to branch the project repository on GitHub and present a modified version of the protocol (likely to be used by experienced developers); or
- to vote for adjustment of any network parameters that do not require protocol modification.

The latter is provided by the system architecture and can be used by all holders of "ENQ". ENQ is the native cryptographic digital token proposed to operate on Enecuum. Following the test period, the voting algorithm is expected to be open for users to present changes to the Enecuum's consensus model. During the test period, the Enecuum team proposes to retain control over the protocol for testing and debugging purposes. Enecuum has been developed using Haskell, a programming language used due to the stability of execution and reduced chances of side effects. A custom version of Cryptonight (Keccak + AES + X11) as the core cryptographic protocol has been chosen because of its high resistance to application-specific integrated circuit ("ASIC") devices [15].

ENQ's are proposed to be generated according to system-specific parameters and paid out to miners as a reward for spending their computational power. Primarily, ENQs can be received and sent with no fees. They can also be used as a payment tool for publishing smart contracts to the network, performing complex mathematical computations on a smart contract, creating custom macroblocks, new Tokens and branches, and participation in PoS mining.

2.3 Potential applications

2.3.1 Initial Coin Offering Platform

The proposed high throughput of the Enecuum blockchain is to allow startups to raise funds at any scale, without the risk of a network hang-up. Hence, initial coin offering ("ICO") participants can be sure they can participate in the ICO and quickly receive their Tokens. Since smart contracts in Enecuum are to be implemented in JavaScript, they will be easy to write for any web developer. Thus the cost of their creation is likely to decrease significantly. Besides, the use of linear logic helps eliminate potential vulnerabilities in smart contract code and helps minimize the risks of attacks.

The "cancellation model" allows issuers to implement complex ICOs with step-by-step raising and return of funds to participants, at any stage of the process. System-specific notation of Tokens, similar to the ERC-20 notation, is intended to simplify entry of the Tokens created on the basis of Enecuum to a cryptocurrency exchange service after the ICO.

Token issuers will be responsible for the appropriate design of their Tokens use cases and ensuring that their Tokens comply with all applicable legal and regulatory obligations.

2.3.2 Infrastructure for financial services and payments

Using Enecuum's "Marks", we aim to enable banks, government agencies and transactional organizations to be able to reliably control targeted spending of received credit and budget funds. The Enecuum infrastructure can also be leveraged to enable secure and efficient payments.

For example, a bank may have a database of customers, which it categorizes on the basis of the nature of their business (construction company, industrial equipment supplier, etc.). The bank has the potential to issue a directed loan in Tokens to a customer having a specific and distinct Mark. The customer will only be able to use these Tokens to pay certain predefined organizations and be able to spend them according to the purpose of the issued loan.

Moreover, the possibility to add an annotation to transactions may, for example, allow for a blockchain-based insurance service that keeps each client's history. The service has the potential to keep user ratings directly on the blockchain and store the information regarding insurance coverage for each user by conducting automatic calculations via smart contracts.

2.3.3 Decentralized storage

The application of sharding technology and the possibility to change transaction duplication parameters allows for effective use of disk space on users' devices. For instance, if four users provide 5 GB of space each and the duplication and sharding parameters are set to 50%, the effective storage capacity for files is 10 GB. Extrapolating this pattern to the entire network, the size of the "global decentralized disk" will grow proportionally preserving the availability of data and a sufficiently high speed of access. This means that in the future users may build on top of Enecuum such services as decentralized hosting, cloud data storage services, and content delivery networks.

We have already developed a preliminary version of the protocol enabling decentralized storage for Enecuum. Generally, the primary goal of the protocol is to provide user Alice (A) with an opportunity to provide some data to user Bob (B) for storage purpose, and to allow A to retrieve this data for a reward. In case some conflicts are present, an arbitrary user (J) should be able to solve the conflict.

The role of J could be delegated to PoS or LPoS node. The main signaling is depicted in Fig. 2.1. Here, arrows represent the communications involving blockchain and dashed arrows correspond to direct ones between the user nodes. Function $hash(x)$ returns the root of the Merkle signature scheme tree [16], as, for example, shown in Fig. 2.2.

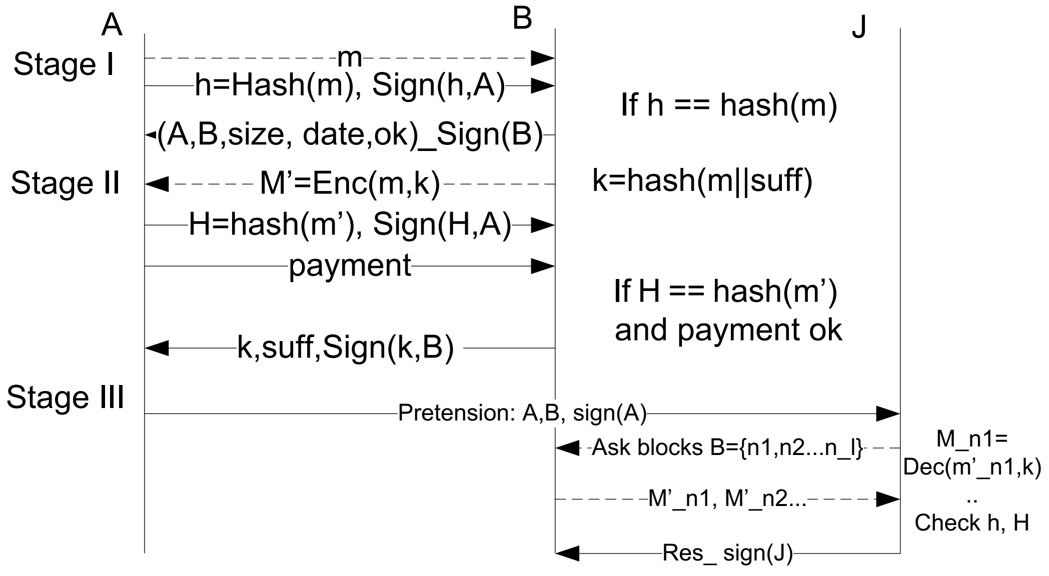

Figure 2.1: Enecuum decentralized storage protocol

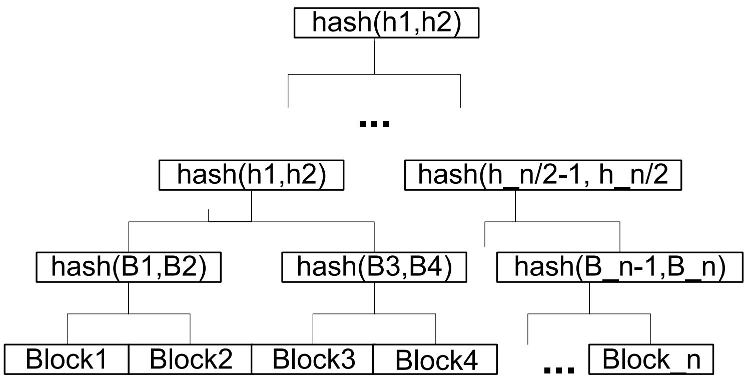

Figure 2.2: Merkle signature scheme tree simplified representation

The protocol could be split into three main phases, explained in Algorithms 1–3. The user data is assumed to be encrypted and is afore-noted as m.

Algorithm 1 Data transmission for storage

```
1: A calculates $h = hash(m)$, where $m$ is it's data.
2: A signs $sign(h, A)$ and adds it to blockchain.
3: A transmits $m$ to B.
4: B calculates $h(m)$ and verifies it with the one signed by A (assessed via blockchain).
5: B accepts $m$ for storage with given limitations (period, amount, etc.) and signs the result.
```

Algorithm 2 Data retrieval

```
1: B transmits $m$, to A in the encrypted way as $(k = hash(m||suff),\ suff = A, B, data_s ize, storage_t ime, random_n once) - m$âĂŹ$ = Enc(m, k)$.
2: A adds $sign(H, A)$, where $(H = hash(m$âĂŹ$))$.
3: A forms a transaction for B based on the storage agreement.
4: If B has successfully received the payment and if the relation between $H$ and $M$‘ is verified – B publishes the key $k$ to the blockchain, thus, making it accessible for A.
5: A retrieves $k$ and decrypts the data.
```

Algorithm 3 Conflict resolution algorithm

```
1: A issues a "ticket" for J concerning the conflict. The ticket is stored in the blockchain.
2: J randomly selects $s$ out of $n$ blocks from $m$ and requests those from B.
3: B returns $h$ and $H$ with all the intermediate hashes to J.
4: J validates $H$ and, in case the validation is successful, checks $h$.
5: In case the validation was successful – ticket is treated as rejected.
6: In case the validation had failed – ticket is treated as accepted, and the transaction is denied.
7: Any decision is signed by J and stored in the blockchain.
```

2.3.4 Microtransactions and IoT applications

The workload on the Enecuum system will increase as the number of users on the Eneecuum platform grows, and decentralized applications are developed on top of the Enecuum blockchain. However, Enecuum proposes to allow for the creation of separate blockchain branches, each with its own consensus rulesets, thus taking the workload off the main system. This, in turn, is to stimulate miners' activity and create conditions beneficial for implementation of microtransaction services.

Enecuum proposes zero transaction fees for a decentralized microtransaction service and very low fees per transaction in case of centralized microtransaction services that involve a large volume of microtransactions from a single wallet. For example, 10,000,000 transactions a day could easily be recorded in several large macroblocks of 10 MB each. The fee is to be calculated per block. Thus it is proposed that there will be extremely low fees per transaction.

We believe this is a perfect use of Enecuum's functionality in relation to the IoT paradigm. An implementation of a simple client for PoA mining on various devices could be able to cover their carried transaction fees completely. Besides, the Enecuum network protocol is designed to provide high availability of such devices by establishing a mesh network between them.

3 Developed protocols and algorithms

3.1 General description

Enecuum is based on PoA algorithm that combines PoW and PoS. This hybrid provides a high degree of network decentralization, while significantly increasing both the network security level and its execution speed. The transaction confirmation process that is proposed to be implemented in Enecuum can roughly be divided into three stages corresponding to the algorithms mentioned above. Note, the technical details are listed after this subsection.

3.1.1 Stage 1

There are two approaches to the first stage. The first approach involves PoW miners finding a proper hash for blocks of varied size, each for its own block, in parallel. After a hash satisfying the current complexity requirements is found, a miner fills the block with transactions and translates it to the network for the second stage involving transaction verification by PoA miners. The second approach is for a PoW miner to find a proper hash, open a macroblock and hold it for a team of PoA miners to fill it with microblocks containing transactions.

3.1.2 Stage 2

During the second stage, PoA miners that are divided into teams act correspondingly to the chosen PoW scenario described in the previous stage. In the case of the first PoW scenario described above, they check the hash in the translated block's header and verify the transactions in the block. In the case of the second PoW scenario, they check the hash in the translated block's header, then create microblocks with transactions, and send them to the macroblock of the PoW miner. Then, depending on the transactions included in the block, PoA miners attach it to one of the system's branches. Checking the block hash for correctness, creating a microblock with transactions and verifying transaction do not require large computational capability, and this operation can be performed even by simple devices, including a mobile phone.

The process of a PoA coalition formation involves calculating a hash to enter the coalition. Each coalition has a significant number of participants and grouped on the base of several parameters, including the node's geographic location, in order to achieve the highest consensus security level.

3.1.3 Stage 3

Details regarding the operation of the third stage are discussed in Section 3.13.

By default, the mining reward is distributed between the participants as follows: PoA – 65%, PoW – 10%, PoS – 25%.

3.2 Network layer

This section provides an overview of the networking strategies utilized by Enecuum, as it is shown in Fig. 3.1. The main system performance metrics are *throughput* and *latency*.

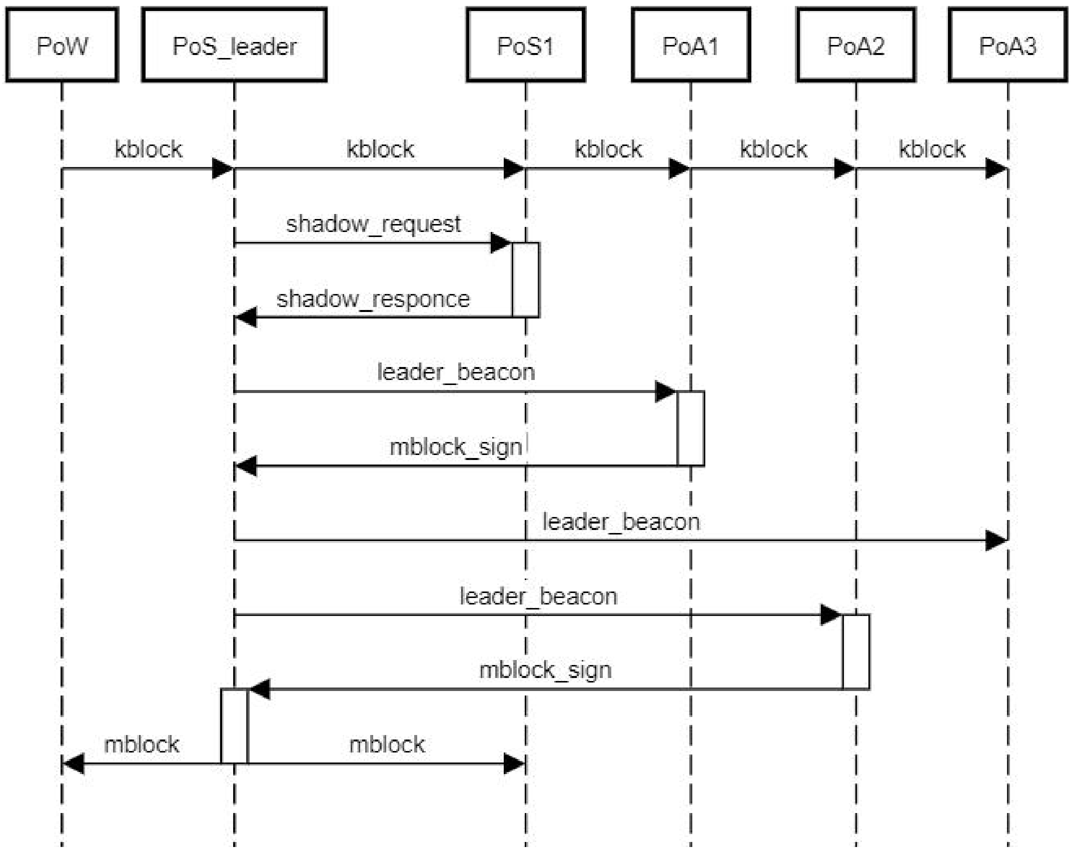

Figure 3.1: Main data signaling

As a reminder, Enecuum has three network nodes in the network namely: solver (PoW), holder (PoS) and publisher (PoA). Solver and Holder nodes are assumed to have 'white' (static) IP addresses and could take part in the actual message routing. The role of publishers are given to constrained devices, such as smartphones and having a static IP address is not a must for them. Therefore, the routing protocol main target is to enable reliable communications between the nodes.

Each solver and holder node also acts as a cashier, i.e., is responsible for processing all k-blocks and microblock in order to operate on top of the local graph data structure. Indeed, the system throughput would be limited by both previously discussed data side in addition to the routing despite the actual network data throughput, disk read/write time, and computational resources.

Enecuum's TestNet has the following main parameters: (i) transaction size – 144 bytes; (ii) microblock size – 100 kb, which corresponds to approximately 650 transactions.

Therefore, the *network throughput limit* corresponds to the results from Table 3.1.

Table 3.1: Link-layer network limitations

Link throughput	**Microblocks per second**
50 Mbps	62
100 Mbps	125
1 Gbps	1250
2 Gbps	2500

The *storage space* limitations depended on the actual hardware and utilized database management system. The database management system utilized in Enecuum TestNet is RocksDB, and all of the k-blocks, microblocks, and transactions are stored there. Most of the storage space is dedicated to transaction-related data. It is expected that market-available SSD disk could process more than 18000 transactions per second [17] that corresponds to 27 microblocks in the context of RocksDB.

The *computational limitations* are not too crucial since most of the cryptographic primitives known today could be executed on mobile devices within relatively small time [18]. Based on the above, the interaction with a database is the bottleneck of the developed system even for cases of relatively low network throughput of just 50 Mbps.

3.3 Routing, throughput, and latency

The routing protocol used in Enecuum TestNet is well-known Chord protocol [19]. Note, the main components used further are detained in 3.4. Namely, solvers and holders, in this case, become the network nodes forming a circle (ring) topology while publishers stay connected with the est via the ring node. Therefore, an overall number of publishers is $n * k$, where n – is the number of network nodes, and k – is a common publisher node connected to the ring node (via TCP sockets). The performance evaluation campaign has shown that the fluctuations in the number of publisher nodes is between one to three thousand nodes.

Relying on on [19] and [20], the unicast or broadcast transmission over the ring topology utilizing Chord protocol requires $log(n)$ hops, where n is the number of nodes in the ring. Based on the signaling strategy, see Fig. 3.1, each microblock publication requires LPoS at least to broadcast two messages, and to receive two acknowledgments and, thus, it is necessary to transmit 4 $log(n)$ messages.

We plan to evaluate the operation of a few hundred ring nodes along with a few thousands of publisher nodes in Enecuum TestNet, and the latency of the network operation is expected to be from 2 to 30 seconds depending on ring nodes count.

The block DAG structure, detailed in 3.5, it is possible to remove the k-block limitation, as discussed in [1]. The k-block generation speed would be selected experimentally after more detailed performance evaluation campaign is executed in Enecuum TestNet.

3.4 Main components

We discuss the main components of the protocol operation as follows.

3.4.1 KBLOCK MESSAGE

kblock message is a broadcast message delivered to all the network nodes. It contains the list of all k-block fields.
$\{kblock\_data\}$

3.4.2 SHADOW_REQUEST MESSAGE

shadow_request message is sent by LPoS node to every available/known PoS nodes. It contains the request for the session key retrieval (all active PoSs already know the current k-block and LPoS ID).

3.4.3 SHADOW_RESPONSE MESSAGE

shadow_response message is sent to LPoS by every PoS in acknowledgment to *shadow_request* message. It contains the secret share calculated by PoS_i based on $hash(kblock\_data)$ and current $LPoS_{ID}$.

3.4.4 SHADOW_KBLOCK

shadow_kblock is the kblock main secret share calculated by LPoS. It is based on $LPoS_{ID}$ and $kblock\_data$ being signed by LPoS secret key as
$\{signature(Secret\ key, \{LPoS_{ID}, kblock\_data\})\}$.

3.4.5 LEADER_BEACON MESSAGE

leader_beacon message is delivered from LPoS to the selected PoA nodes stating the fact of leadership of current k-block.
$\{hash(kblock\_data), signature(Secret\ key, LPoS_{ID}, kblock\_data)\}$.

3.4.6 MBLOCK_SIGN MESSAGE

mblock_sign message is sent to LPoS from PoA after $\{signature(Secret\ key, \{LPoS_{ID}, kblock\_data\})\}$ is verified.

3.4.7 MBLOCK MESSAGE

mblock message is a message broadcasted by LPoS after PoAs have delivered the required number of microbocks to LPoS.

3.5 Directed Acylic Graph (DAG)

3.5.1 General background

The moment there are enough pending transactions to start assembling a block, the block creation process begins. Analyzing specified parameters of each transaction, miners determine its value for the system and add it into a corresponding block.

In Enecuum, the block size is not proposed to have a fixed value and may vary from 4 KB to 4 MB. 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 into a macroblock, thereby allowing the storage of large volumes of data on the blockchain.

Bitcoin-NG protocol is proposed to be introduced into Enecuum macroblocks [21] 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, we do not have to wait until an entire macroblock is completed, its hash is found, and it is synced between all nodes on the network – small microblocks can be generated concurrently inside it.

The block structure consists of 3 main sections represented in the Figure 3.2.

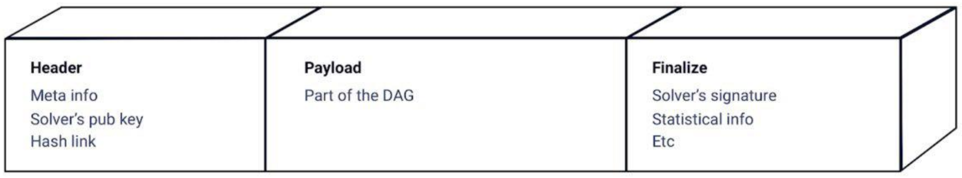

Figure 3.2: Block structure in Enecuum

The main reason behind the utilization of this protocol lies in its possibilities to increase the mining speed in the system, i.e., to increase the number of blocks generated by the system within the selected time frame. The fundamental limit of which is the distribution time of the newly generated block through the system. In case the generation time is smaller – the probability of forking in two distant sections of the network may arise tremendously. DAG is expected to allow the addition of new blocks in different network segments without forking.

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

- Graph construction and 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 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.

3.5.2 Proposed utilization of DAG

First, we define the graph walk procedure. We start 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 [22].

We assume that there exists the deterministic algorithm allowing to calculate 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 i:

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 $difficulty$.

3.6 Target number of transactions

Commonly, the blockchain implies to utilize just one 'chain' of blocks limited by the number of transactions per second due to overall system limitations (not dependent on the number of PoAs). Basically, we have a coalition of PoS nodes actually operating with the blockchain. We propose to utilize parallel chains served by different PoSs and thus improve the overall system performance.

For example, consider a simple case where a coalition of PoSs (PoS_i, PoS_j, PoS_k) are serving chain C_A while a coalition of PoSs (PoS_x, PoS_y, PoS_z) are serving chain C_B independently. Which, in a broader sense, may be operational as two standalone blockchain structures without any way of interoperability without involving the exchange center. In Enecuum, nodes from different branches (C_A and C_B) can still interoperate since all PoSs have their secret keys generated from the same main system secret. Thus, any user operating in C_A would have an opportunity to transfer its funds to C_B based on the algorithms listed further in subsection 3.7 in a trusted and anonymous way.

Enecuum would allow using a different number of serving chains dynamically adjusting to the system load. The challenge here is additional delay brought by the transactions between different serving chains as a trade-off to the required transaction number in the overall system. The second challenge is transaction verification by users in different serving chains since it brings additional need to local storage and time for checking different blockchains, which is solved by more frequent blockchain genesis points redistribution and validation based on initial secret shares calculated from main system secret.

3.7 Anonymous transaction realization

The primary goal for the transaction anonymization is related to hide the payer from identification during the transaction and in the future. Payer personal data should be hidden from the payment recipient and other blockchain participants.

In order to achieve this goal, there is a need to create a separate 'check,' which would not be bounded either with its owner nor associated/tracked via blockchain. At the same time, check validity should verifiable by any involved party.

A check is formed by using blind Chaum signature [23]. Here, each PoS must have a set of generated keys for signing checks of different denominations. Values and number of denominations should be fixed.

3.7.1 Anonymous check issuing

1: A send n tokens to PoS_i to purchase the anonymous checks.
New transaction is created: $n, A \rightarrow PoS_i, E(chk), hash(chk_j)$, where chk_j is a salted anonymous check calculated as $chk_i = n, ID(chk_j)$; $E()$ is a reversible for A and PoS_i function with session key known only to them. The number of checks per transaction may vary and the more checks are requested – the higher creation fee is added. Higher number of checks do not allow to track the transaction based on the number of tokens.

2: Next, PoS_i sends singed anonymous check to A via blockchain as
$E((chk_j), sign(PoS_i)), hash((chk_j), sign(PoS_i))$

3: A verifies the validity of received check, removes the salt from $ID(chk_j)$ and forms the final check as $Chk_j = n, ID(chk_j), sign_{PoS_i}$

4: In case check is not valid, A publishes the secret key used for the check encryption and the transaction is denied if PoS_i signature is incorrect.

Note, PoS_i should fix on his balance doubled amount of requested tokens during the anonymous check creation. These funds should be available when PoS is offline: in Ledger or through a smart contract. Therefore, a track of issued checks is kept. In this case, when there is not enough money on the PoS_i account – it cannot anonymize tokens.

3.7.2 Anonymous payments

1: Each issued check could be used for payment from A to B
Here, B first verifies the check's signature. Second, B verifies that this check was never used before via blockchain. Next, B creates a transaction stating the reception of the

check and receives the funds.

2: In case there are more checks per PoS_i than available funds on the corresponding wallet, all related checks are blocked, and all the anonymized checks' related funds are returned to owners. Note, it is impossible to find out which of the owners has already spent their checks. The entire returned funds become a penalty for PoS_i, which is made to prevent uncontrollable emission by any PoS.

In order to improve the level of anonymity, each user is recommended to purchase checks with different amounts and from different PoSs, which would make post analysis more complex.

3.8 Ledger

The following parameters are considered during the ledger calculation: k-block mining; reward for microblock publishing; and transaction fee. The rewards are dynamic and based on the blockchain operation history. The main functions responsible for that are: $reward_K()$ and $rewards_m()$ calculating the rewards for each system state, see subsection 3.12 for details.

The k-block arrangement algorithm utilized in Enecuum is DAG, see subsection 3.5. The algorithm for arranging the microblocks in k-block is described in subsection 3.5. 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.

For example, consider the following structure:

```
1: $A$ mined k-block with data:
2:     $D$ published microblock with data:
3:         $A$ sent to $B$ 10 coins;
4:         $B$ sent to $E$ 3 coins;
5:         $C$ sent to $E$ 2 coins.
6:     $E$ published microblock with data:
7:         $A$ sent to $D$ 1 coin;
8:         $D$ sent to $E$ 1 coin;
9:         $A$ sent to $C$ 25 coins.
```

In this case, the following would be executed:

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.

Table 3.2: Example ledger

Action	**Ledger status**
Assign A $reward_K()$ coins	$\{A:100\}$
Assign D $reward_m()$ coins	$\{A:100, D:10\}$
Transaction from A to B 10 coins	$\{A:90, B:10, D:10\}$
Transaction from B to E 3 coins	$\{A:90, B:7, D:10, E:3\}$
Transaction from C to E 2 coins – **invalid**	$\{A:90, B:7, D:10, E:3\}$
Assign E $reward_m()$ coins	$\{A:90, B:7, D:10, E:13\}$
Transaction from A to D 1 coin 1	$\{A:89, B:7, D:11, E:13\}$
Transaction from B to E 3 coins	$\{A:89, B:7, D:10, E:14\}$
Transaction from A to C 25 coins	$\{A:64, B:7, C:25, D:10, E:14\}$

3.9 Rewarding policy

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.

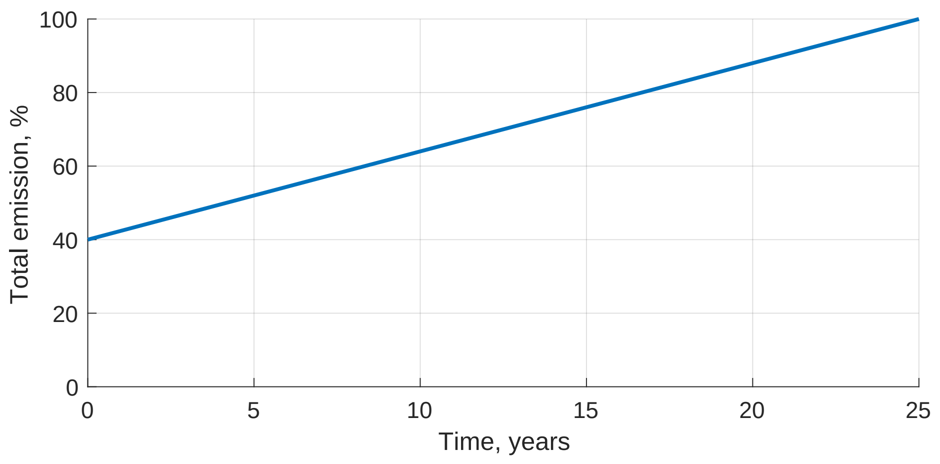

Figure 3.3: Emission curve.

The emission distribution balance is a dynamic system property and could be used as a tool to mitigate malicious activity between different nodes. The current emission curve is described as $E(x) = 2(x) + 40$.

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.

3.10 DIFFICULTY

The available details are provided in subsection 3.4.

3.11 SMART CONTRACTS

Smart Contracts in Enecuum are to be written in JavaScript and executed on Google's V8 engine.

3.11.1 LIGHT (LOGICAL) SMART CONTRACTS

Those contracts are to be composed exclusively of mathematical formulae and based on the business-oriented SHARNELL-like linear logic. Linear logic is entirely predictable, hence minimizing the chance of any potential vulnerability.

Logical, smart contracts are to consist of a "data card" containing conditions and parameters, and the formula itself which takes into account these conditions and parameters with the possibility of full or partial achievement and actuating. Each condition of a logical contract is to be placed in the data card and assigned a corresponding symbol. Later, a mathematical formula fully reflecting the terms of the contract is created. The Π-calculus system is used to ensure computations are run in parallel. This type of smart contract is ideal for performing the most common operations and transactions, such as multisig, escrow and so on.

3.12 AI-BASED DYNAMIC DIFFICULTY CALCULATION

Mining a blockchain network such as Enecuum, is a growing and open market system with strong external influences (marketing, new technologies, economic and legislative changes, to name a few). Because of this, it is practically impossible to create an algorithm to predict the required difficulties and rewards with 100% accuracy. In EnecuumâĂŹs case Proof of Work, Proof of Activity and Proof of Stake all have partial interaction and competition with each other, making our market interactions an order of magnitude more complex than the more straightforward single product market of a Proof of Work-only blockchain. Such a blockchain typically uses a variation on a time-series algorithm, ranging from moving averages to specialized algorithms.

Doing away with fixed algorithms allows for more flexibility and market-oriented behavior. For example, in Enecuum it is no longer required to have a fixed emission scheme; the emission behavior can be tuned to the actual network requirements.

The Enecuum reward and difficulty system, Neuro, is a custom designed a neural network to fit the requirements of a modern blockchain. Neuro uses historical data, statistics, stored in the blockchain itself 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.

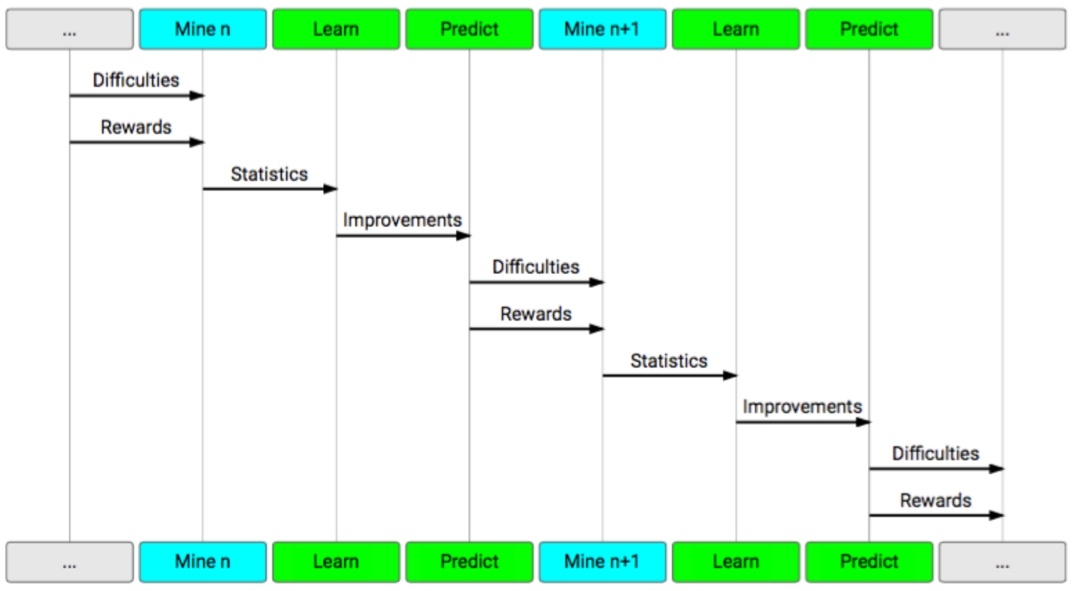

Figure 3.4: Enecuum AI.

In a plain, non-recurrent neural network each forward cycle starts with a clean state, 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.

3.13 TRINITY

The following subsection describes the interaction between nodes during the blockchain construction (see structure in subsection 3.5). Subsection 3.2 provides an overview of the communications side of the system allowing the nodes to communicate in peer-to-peer, broadcast and multicast ways.

The system is based on three types of users: (i) solver (PoW); (ii) holder (PoS); and (iii) publisher (PoA). None of those could act as 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.

3.13.1 SOLVER (PROOF-OF-WORK)

PoW solver is responsible for the generation of new k-blocks.

Main requirements: (i) access to the Internet; (ii) storage (required to store the blockchain structure); (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.

Conventional Nakamoto consensus 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. k-block contains its solver's public key . The rewards are calculated dynamically according to subsection 3.12.

The selection of the hashing function does not affect the overall system operation directly. The TestNet utilized for performance evaluation uses SHA-256, but this choice is temporal since modern ASICs can easily calculate it.

3.13.2 Holder (Proof-of-Stake)

PoS holder is a node holding a reasonable amount of coins. The requirements for becoming a holder are described in subsection 3.12. The node can prove hive eligibility to become a holder based on the protocol described in subsection 3.14.1.

Key SK_{HK} is a shared key distributed between a set of holders based on the Lagrange interpolation formula [24]. The corresponding PK_{HK} is known to any node. The *Resident* node (described in subsection 3.13.4) 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 3.14.2 to verify who has the right to distribute the publication keys during this system operation state. This interval is set to 100 k-blocks in TestNet. The Leading PoS (LPoS) selection results are then stored as statistic blocks (see subsection 3.5) and may be verified using the protocol described in subsection 3.14.3.

Next, LPoS is generating the publication secret key after the k-block retrieval. The publication public key is calculated based on the k-block ID (hash sum). The secret key and the corresponding shares are calculated based on the protocol described in subsection 3.14.4. The shares are then distributed to PoA publishers selected for the publishing of the microblocks related to this k-block. 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 publication key generation process.

3.13.3 Publisher (Proof-of-Activity)

PoA publishers are involved in the microblock publishing process. Each microblock should be verified by a coalition of grouped PoA publishers.

In order to retrieve a new k-block, PoA publisher is requesting the LPoS the key share correlated with this k-block. Next, PoA generates the microblock payload (array of transactions) and forwards it to the other PoAs in the coalition. After the necessary number of PoAs have signed the payload (according to threshold schema). Therefore, the microblock data becomes validated by k PoA nodes in the system, and their participation may be verified later on The PoA rewards are based on participation in the verification procedure.

3.13.4 Resident

The resident is one of the PoS holders of the system being controlled by Enecuum. This 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 3.14.9 and 3.14.10. 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.

3.14 Cryptographic protocols

Each k-block has it's own unique $IDk - block\ number$ calculated by $IDk - block\ number = f(k - block\ data)$ and $f()$ is a hashing function.

3.14.1 Stakes verification protocol

The protocol represents the phase while any participant is proving his actual stake to another one.

Main requirements: (i) possibility of verification; (ii) resistance against forgery. Input data: (i) size of data; (ii) actual difficulty.

Operational challenges: The protocol has two versions, for computationally resource non-constrained and constrained devices.

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

Main requirements: (i) resistance against the repetitive selection of the same miner during a series of sessions, i.e., improved randomization; (ii) 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.

Input data: size of data; number of transactions.

Operational challenges: (i) The appropriate location for the protocol execution-related data per session, i.e., participants, selection rules, etc.).

1. Stage A: After new k-block is published, all the potentially involved PoS miners advertise themselves.

2. Stage B: Every voting procedure participant stores the list of the PoS candidates locally in case the signature was validated correctly. The list is then arranged in the lexicographical order.

3. Stage C: After the list is constructed from N nodes, each participant calculates the hashing function $r = \frac{Hash(k-blockId|PoS_1|...Pos_N)}{Hash_{max}} \in (0,1)$.Therefore, the voting is further based on r and comparing it to newly generated discrete random variable in the same bound. Therefore, 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 it's interval.

4. After the tagged PoS was selected (LPoS status), it transmits the request the key to the rest PoSs nodes and after he receives at least k of replies (basically, those have the same list on their side), the secret key, to be used in protocol 3.14.4 is generated.

5. Stage D: Execution of the protocol 3.14.4.

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

3.14.3 Leading PoS miner verification protocol

In Progress

3.14.4 Leading PoS miner key generation for PoA verifiers protocol

Main requirements: (i) keys should have a property of single-use; (ii) keys should be distributed securely; (iii) keys could not be generated by any user; (iv) keys do not contain any information related to PoS miner secret keys.

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

$$ID_k = f(block_k), \tag{3.1}$$

where f is a hashing function SHA-1,2,3.

The protocol execution could be done in case the leading miner is selected by $LPoS = PoS_i$ according to the protocol described. Each PoS has its own 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} may be selected as PK_{LPoS} or a function of this key . PK_{LPoS} could be thus selected as

$$PK_{LPoS} = f(block_k || ID_{LPoS} || L_{voting}), \tag{3.2}$$

$$SK_{LPoS} = newSK_{PoA}, \tag{3.3}$$

where L_{voting} is a list of PoS miners that participated in the process.

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

3.14.5 Protocol of the PoA applicability for microblock generation procedure

The coalition of PoA miners is selcted after new k-block is published. It is selected based on constant N_{PoA} per node based on the corresponding ID so 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 brootforcing of the ID is a computationally complex task.

Algorithm 4 Initialization of ID-based schema with distributed PKG

```
1: Define groups:
2:         Define $G1$ is a cyclic group of order $q$ (number of points on elliptic curve);
3:         Define multiplicative group $G2$;
4: Define functions:
5:         $H1 : (0,1)* \rightarrow G1$;
6:         $H2 : G2 \rightarrow (0,1)*$;
7:         $H3 : (0,1)* \rightarrow Zq$;
8:         $e : G1 \times G1 \rightarrow G2$ (Pairing);
9: Define MasterSecretKey $s \in Zq$;
10: **Define** $P$ :generator of $G1$;
11: **Define** Public MasterPublicKey= $sP$
```

Algorithm 5 PKG (k, n) MasterPublicKey splitting

```
1: Generate random polynomial is residue field $q$: $deg(f(x)) = k - 1$;
2: Each participant receives its key share of MasterPublicKey $ss_i = f(id_i) mod\ q$.
```

Algorithm 6 Session key SK_{LPoS} generation for $LPoS$

```
1: Each $k$ participants calculates $PK_{LPoS}$ according to equation 3.2.
2: Transmits its $ss_i$, $PK_{LPoS}$ and $id_i$ to LPoS.
```

Algorithm 7 Secret key recovery

```
1: LPoS is calculating $SL_{LPoS}$ based on the received from algorithm 3.14.4 data as
2: $SK_{LPoS} = \sum_{i=1}^{k} \lambda(id_i, 0) ss_i\ PK_{LPoS}$, where $\lambda(id_i, 0)$ is a Lagrange coefficient generated
   per coalition for each user $id_i$ and 0.
```

3.14.6 Generation of microblock by PoA for current k-block

Main requirements: (i) simultaneous and independent execution of the coalition members; (ii) the data exchange minimization; (iii) in-block additional data minimization; (iv) confirmation of the participation in the verification process.

Input data: block; key retrieved from PoS, own secret key.

Operational challenges: Resistance against double transactions. Block generation period threshold selection.

Each PoA miner verifies if it is applicable for new microblock geenration 3.14.5 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 with PoA SK and immediately published.

3.14.7 LPoS microblock assurance protocol

After protocol 3.14.2 is executed and new session key is generated 3.14.4, LPoS starts to assure the microblocks.

1. 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.
2. Stage B: LPoS verifies the validity of transactions in the microblock based on the ledger.
3. Stage C: In case the verification is positive, each microblock is signed with SK_{LPoS} from protocol 3.14.4 according to:

Algorithm 8 Microblock signature protocol

```
1: LPoS generates $r$ from $Zq$;
2: Calculates $R = rP$ and $S = SK_{LPoS} + rH1(ID_{LPoS}, M) = sQ + rH1(ID_{LPoS}, M)$, where $M$
   is the entire microblock;
3: 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.

3.14.8 Cryptographic microblock verification protocol

Main requirements: (i) should be executable at any node; (ii) should be based only on publicly available information; (iii) a possibility of the status check of any wallet.

Input data: block; key retrieved from PoS, own secret key.

Operational challenges: should microblock be verified without the entire blockchain evaluation?

The signature of the microblock is based on the R and S pair are verified based on:

$$e(P, S) = e(MPK, PK_{LPoS} = Q) \cdot e(R, H1(ID_{LPoS}, M)), \tag{3.4}$$

where $MPK = sP$.

3.14.9 Distributed PKG secret update protocol

This protocol is executed either whenever the PoS miners set is changes, or during the ledger recalculation when any of the PoS nodes loses the PoS status.

New key shares are distributed by Resident node. This node 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.

3.14.10 Distributed PKG new secret share transmission protocol

3.15 Direction of work

- Hashing function selection making the ASIC mining computationally difficult;
- Publishers and reward parameters selection so that the publisher emulation would become inefficient on personal computers.

4 System evaluation

In progress. Description of TestNet.

4.1 System evaluation

4.1.1 Performance evaluation of LPoS load concerning k-block processing time

In this subsection, we provide an example of the system operation evaluation from communication (signaling) perspective since the computational analysis is not commonly used. The main focus is given to 'tagged' LPoS and the packet transmission time between related nodes (according to BlockDAG) and the corresponding packet processing and storing (interaction with the database) metrics.

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 [25]. The broadcast procedure is balanced according to [26].

The message sizes utilized in this campaign are: microblock – 100kb ($\tilde{6}$50 transactions); others – 144 bytes ($\tilde{1}$ transaction). Table 4.1 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.

Table 4.1: Approximate load brought by different message type

Type	**Source**	**Destination**	**Other Chord nodes**	**Average number of other Chord nodes**	**Example:** $n = 300$
$A \to S$	1	1	-	-	-
$S \to A$	1	1	-	-	-
Chord → Chord	1	1	$< log(n)$ (1/1)	$((log((n))/2 - 2))/((n-2))$	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	$1/\dots$	$1/\dots$

Next, we focus on the packet propagation time faced by our system. The use of TCP for our system generally increase the Round Trip Time (RTT)/delay in a tradeoff to reliability. The approximations used in this campaign are based on the public data[2,3]. Note, the potential

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

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

higher delays faced by the cellular network users are not expected to affect our system operation. Table 4.2 provides example values of RTTs between well-known key locations.

Table 4.2: Example delays between known data-centers

	Amsterdam	Oakland	Bangalore	Kishinev	London	Los Angeles	Moscow	New York	Paris	Tokio
Amsterdam	-	298	158	46	9	136	50	85	20	242
Oakland		-	271	349	267	179	348	251	291	186
Bangalore			-	196	170	241	173	202	163	115
Kishinev				-	56	172	60	129	55	297
London					-	131	54	71	4	227
Los Angeles						-	183	69	145	108
Moscow							-	126	54	207
New York								-	74	210
Paris									-	226
Tokio										-

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 and example measurements for metropolitan area are given in Table 4.3. Results covering smartphones are given in Table 4.4.

Table 4.3: Measurements of the delays between personal computer and cellular operators servers

	Vimpelcom ltd	Rostelecom	PJSC MegaFon	Forest Net	ITSP Prometey
PC	1	21	22	19	18
WiFi 1	2	21	23	20	19
WiFi 2	25	23	8	17	18

4.1.2 System Level Simulation

For our performance evaluation campaign, we selected project *p2psim* mainly because it has an opportunity to emulate the Chord. Moreover, it has a set of real packet propagation measurements between thousands of nodes, collected in *kingdata* package. Next, we propose to utilize many simultaneous TCP sockets between the nodes in the link throughput allows it. By this means, the delay may be significantly decreased. We detail our system model in the following.

Here, 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

Table 4.4: Measurements of the delays between PCs and smartphones and cellular operators servers

	PC	Smartphone
Argentina	28-59	-
Australia	1-62	12-27
India	3-255	16-638
Italy	1-91	5-94
Japan	4-57	48
Pakistan	1-207	19-94
Russia	1-96	3-80
Singapore	1-225	-
United States	1-156	9-51
Vietnam	3-27	16-65

analyzed the number of signaling messages required for different messages dissemination, and the results are shown in Table 4.5

Previously, we have analyzed the simplified operation of the system. 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 are presented in Table 4.6. 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. 4.1. Note, $lat_{ch} = 0.150$ seconds, so that 30 seconds equals 200 timeslots.

4.2 Potential Attacks and Limitations

The scope of our work includes security analysis in respect of both the network and blockchain.

4.2.1 Network-related challenges

Most existing consensus-building mechanisms utilized by distributed ledgers present a trade-off between a large number of transactions per second and degree of network centralization. Thus, the desire to increase the number of processed transactions often leads to growing risks associated with the system reliability. Besides, as the size of a blockchain grows, it requires more disk space, a stronger Internet connection, and higher computational power. All this may result in a decreasing number of full nodes and have a negative impact on the security of the entire network.

Generally, the developed system is a representative example of a distributed network and, thus, it is affected by a wide range of P2P related attacks. Next, we shortly discuss the most essential ones.

First, a Sybil attack represents a case where a node or other single entity on a network presents multiple identities to other nodes. When the amount of identities is high enough,

Table 4.5: Description and and number of messages needed for blocks dissemination

Time	**LPoS**	**Chord's nodes**	**PoAs**	**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
$5 * lat_{ch}$	$shadow_{request}$ transmitted			
$13 * lat_{ch}$	Received $shadow_{response}$	k-block received	k-block received by the majority	LPoS is ready to send $leader_{beacon}$
$23 * lat_{ch}$		The remaining ones are transmitting $leader_{beacon}$ to PoAs	Receiving $leader_{beacon}$	LPoS's disk utilization increases. $mblock_{sign}$ arrival begins
$33 * lat_{ch}$	Majority received			LPoS disk load is still present
$45 * lat_{ch}$	Last $mblock_{sign}$ received			LPoS starts to broadcast mblock via Chord.
$54 * lat_{ch}$		mblock received		The procedure is over. Total time is around 8 seconds.

capable of taking over the network, a disadvantage of a decentralized blockchain system is its limited bandwidth. Unique node identifier should be utilized to overcome this issue. However, no ultimate solution could be found to overcome this challenge.

Another attack is directly related to retrieving the same identifiers by different nodes in the system that may drastically influence the routing. The challenge could be potentially solved by randomization of the identifiers during the system operation period.

Table 4.6: Description and and number of messages needed for blocks dissemination: pessimistic approximation

Time	**LPoS**	**Chord's nodes**	**PoAs**	**Action**
0				PoW's k-block transmission
$10 * lat_{ch}$	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. For example, $0.05 * k * lat_{ch}$.
$11 * lat_{ch}$	$shadow_{request}$ transmitted			
$31 * lat_{ch}$	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.
$41 * lat_{ch}$		The remaining ones are transmitting $leader_{beacon}$ to PoAs	Receiving $leader_{beacon}$	**If $k \leq 600$ – k-blocks are already delivered to PoA**
$43 * lat_{ch}$	Majority received			LPoS disk load is still present: $k * n * msig_{size}/disk_{speed}$ equals $k * n/180000$ seconds
$(43 + x) * lat_{ch}$ or $(20 + y) * lat_{ch}$	Last $mblock_{sign}$ received			Either the disk utilization is finished, or k-block distribution is finished.
$(+10) * lat_{ch}$		mblock received		The procedure is over. Total time is around 8 seconds.

5 Conclusions

The main protocol objectives:

- As in any blockchain system, Enecuum should check the participants' work postfactum.
- Also ENQ tried to maximally divide the system function where one is mining, second one is publishing the microblocks and the third one is generating the keys.
- To complicate the system "compromizing".
- To involve the mobile users to the system.

Currently, the plans of Enecuum are:

- Evaluate the possible quantity of different intentional devices;

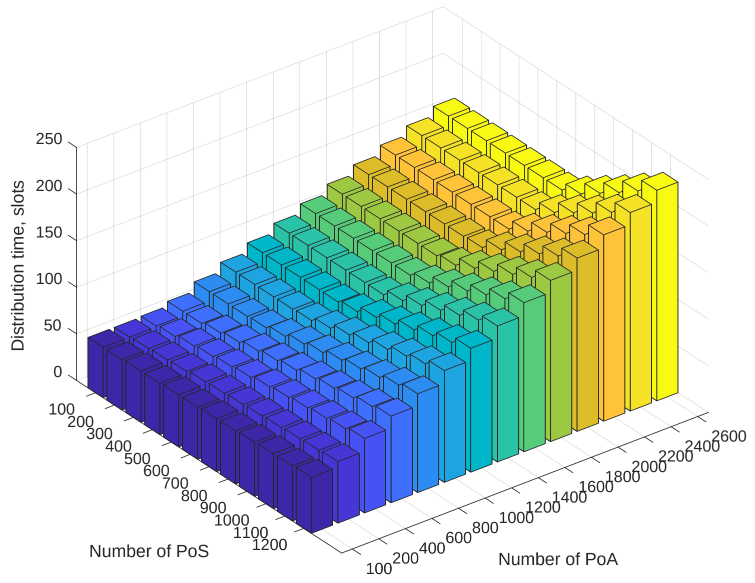

Figure 4.1: System operation time varying number of PoSs and PoAs

- Engage cryptography specialists to analyse the algorithm
- Make an evaluation of a system output capacity using the methods of simulation modeling.
- Implement the algorithm in Testnet.

So, evidently, Enecuum is going forward and also looking for the cryptography specialists who would like to share their experience with the project and become a part of future.

References

[1] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. (2008)

[2] Christidis, K., Devetsikiotis, M.: Blockchains and smart contracts for the Internet of Things. IEEE Access **4** (2016) 2292–2303

[3] Frey, D., Makkes, M.X., Roman, P.L., Taïani, F., Voulgaris, S.: Bringing secure bitcoin transactions to your smartphone. In: Proceedings of the 15th International Workshop on Adaptive and Reflective Middleware, ACM (2016) 3

[4] Vukolić, M.: The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In: International Workshop on Open Problems in Network Security, Springer (2015) 112–125

[5] VNI Cisco: Global mobile data traffic forecast 2016–2021. White Paper (2017)

[6] Bentov, I., Lee, C., Mizrahi, A., Rosenfeld, M.: Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake [extended abstract]. ACM SIGMETRICS Performance Evaluation Review **42**(3) (2014) 34–37

[7] Xiong, Z., Feng, S., Niyato, D., Wang, P., Han, Z.: Optimal pricing-based edge computing resource management in mobile blockchain. In: 2018 IEEE International Conference on Communications (ICC), IEEE (2018) 1–6

[8] King, S., Nadal, S.: Ppcoin: Peer-to-Peer Crypto-currency with Proof-of-Stake. self-published paper, August **19** (2012)

[9] Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS) **4**(3) (1982) 382–401

[10] Back, Adam et al.: Hashcash-a denial of service counter-measure (2002)

[11] Eyal, I., Gencer, A.E., Sirer, E.G., Van Renesse, R.: Bitcoin-ng: A scalable blockchain protocol. In: NSDI. (2016) 45–59

[12] Shamir, A.: Identity-based cryptosystems and signature schemes. In: Workshop on the theory and application of cryptographic techniques, Springer (1984) 47–53

[13] Cocks, C.: An identity based encryption scheme based on quadratic residues. In: IMA International Conference on Cryptography and Coding, Springer (2001) 360–363

[14] Shamir, A.: How to Share a Secret. Communications of the ACM **22**(11) (1979) 612–613

[15] Van Saberhagen, N.: Cryptonote v 2.0 (2013)

[16] Xu, J., Wei, L., Zhang, Y., Wang, A., Zhou, F., Gao, C.z.: Dynamic fully homomorphic encryption-based merkle tree for lightweight streaming authenticated data structures. Journal of Network and Computer Applications **107** (2018) 113–124

[17] Marotto, F.: Performance Benchmarks. `https://github.com/facebook/rocksdb/wiki/Performance-Benchmarks` (November 2018)

[18] Ometov, A., Masek, P., Malina, L., Florea, R., Hosek, J., Andreev, S., Hajny, J., Niutanen, J., Koucheryavy, Y.: Feasibility characterization of cryptographic primitives for constrained (wearable) IoT devices. In: Proc. of PerCom Workshops. (2016) 1–6

[19] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on Networking (TON) **11**(1) (2003) 17–32

[20] Kaija, K.: The implementation and performance of Chord. PhD thesis, Helsingin yliopisto (2018)

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

[22] Kahn, A.B.: Topological sorting of large networks. Communications of the ACM **5**(11) (1962) 558–562

[23] Chaum, D.: Blind Signatures for Untraceable Payments. In: Advances in cryptology, Springer (1983) 199–203

[24] Ometov, A., Zhidanov, K., Bezzateev, S., Florea, R., Andreev, S., Koucheryavy, Y.: 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)

[25] Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOMM Computer Communication Review **31**(4) (2001) 149–160

[26] Huang, K., Zhang, D.: Dht-based lightweight broadcast algorithms in large-scale computing infrastructures. Future Generation Computer Systems **26**(3) (2010) 291–303

6 Additional information

This paper and any other documents published in association with this paper relate to the intended development and use of the Enecuum platform ("Enecuum"). They are for information purposes only and may be subject to change.

• **This paper describes a future project**

This paper contains forward-looking statements that are based on the beliefs of Enecuum HK Limited, a Hong Kong incorporated company (CR: 2562183) ("Company"), as well as certain assumptions made by and information available to the Company. Enecuum, as envisaged in this paper, is under development and **is being constantly updated**, including but not limited to essential governance and technical features. The ENQ token ("ENQ") involves and relates to the development and use of experimental platforms (software) and technologies that may not come to fruition or achieve the objectives specified in this paper.

If and when Enecuum is completed, it may differ significantly from the network set out in this paper. No representation or warranty is given as to the achievement or reasonableness of any plans, future projections or prospects and nothing in this document is or should be relied upon as a promise or representation as to the future.

• **No offer of regulated products**

ENQ is not intended to represent a security or any other regulated product in any jurisdiction. This document does not constitute an offer or solicitation of securities or any other regulated product, nor a promotion, invitation or solicitation for investment purposes. The terms of the purchase are not intended to be a financial service offering document or a prospectus of any sort. ENQ does not represent equity, shares, units, royalties or rights to capital, profit, returns or income in the platform or software or the Company or any other company or intellectual property associated with the platform or any other public or private enterprise, corporation, foundation or other entity in any jurisdiction.

• **This paper is not advice**

This paper does not constitute advice to purchase ENQ. It must not be relied upon in connection with any contract or purchasing decision.

• **Risk warning**

The purchase of ENQ and participation in Enecuum carries with it significant risks. Before purchasing ENQ, you should carefully assess and take into account the risks, including those listed in any other documentation.

• **Views of the Company**

The views and opinions expressed in this paper are those of Enecuum and do not reflect the official policy or position of any government, quasi-government, authority or public body (including but not limited to any regulatory body of any jurisdiction) in any jurisdiction. The information contained in this paper is based on sources considered reliable by the Company, but there is no assurance as to their accuracy or completeness.

• **English is the authorized language of this paper**

This paper and related materials are issued in English only. Any translation is for reference purposes only and is not certified by the Company or any other person. No assurance can be made as to the accuracy and completeness of any translations. If there is any inconsistency between a translation and the English version of this paper, the English version prevails.

- **No third party affiliation or endorsements**

References in this paper to specific companies and platforms are for illustrative purposes only. The use of any company and/or platform names and trademarks does not imply any affiliation with, or endorsement by, any of those parties.

- **You must obtain all necessary professional advice**

You must consult a lawyer, accountant, tax professional and/or any other professional advisors as necessary prior to determining whether to purchase ENQ or otherwise participate in the Enecuum network.