# Proof of Usage: User-centric consensus for data provision and exchange

Samuel Masseport, Jorick Lartigau, Benoit Darties, Rodolphe Giroudeau

# Proof of Usage: User-centric consensus for data provision and exchange

Samuel Masseport[1,2], Jorick Lartigau[1], Benoît Darties[2] and Rodolphe Giroudeau[2]
[1] Pikcio SAS, Montpellier, France
{samuel.masseport, jorick.lartigau}@pikcio.com
[2] University of Montpellier, LIRMM, France
{samuel.masseport, benoit.darties, rodolphe.giroudeau}@lirmm.fr

***Abstract*— This paper describes the Proof of Usage, a consensus algorithm empowering a new paradigm to incentive usage, valuation and control of user data. PoU is introduced in the scope of permissioned blockchain, designed for a user-centric personal data marketplace that is compliant with today's regulations, which public blockchains cannot satisfy. Previous consensuses such as Proof of Work and Proof of Stake do not encourage coin spending and usage (proof of stake does the opposite) despite the value of the currency depending on its use. Our paper begins with a contextualization of blockchain and the consensus. Thus, we discuss the motivation towards a new model for personal data exchange in a decentralized yet controlled environment where the PoU consensus interprets transactions and user control over their data. Finally, the paper describes the protocol and process flow, the two different approaches regarding the rewarding mechanism and the security measures that PoU ensures.**

## I. CONTEXT

In 2008, Satoshi Nakamoto introduced Bitcoin [13], the first fully decentralized cryptocurrency. To this day, Bitcoin is the most widely adopted and reliable economic system built on a peer-to-peer network, enabling online payments directly from one party to another without a sole fiduciary entity. This was the starting point for Blockchain technology [16]. The Bitcoin consensus is based on a Proof of Work (PoW) protocol [6, 7] that protects the network from Denial of Service (DoS) attacks and double spending. Nodes compete to add a block to the current chain (named "miner") when trying to solve a hard asymmetric mathematical problem[1]. Bitcoin blockchain rewards the miner when they add a block to the blockchain. This compensation encourages miners to use their computing power to improve the defense of the network. However, PoW has two mains issues i.e. the need for a significant amount of electrical power and its limited scalability in terms of transaction processing (around seven per second).

As Bitcoin was often criticized for its high energy consumption due to the fierce competition to add a block, a new consensus protocol was proposed, i.e. the Proof of Stake (PoS). It was introduced by Sunny King and Scott Nadal in 2012 [8]. Rather than using the computing power of miners to add blocks to the current chain and secure the network, Sunny King and Scott Nadal defined an alternative method called "staking" where a deterministic algorithm chooses a network participant, i.e. a node. The selection is based on the number of coins in each node's possession or said "stack". For example, if there are one hundred coins on the network, a node holding ten coins will have 1 in 10 chances to be the selected block miner and earn the reward. Nodes holding ten coins are ten times more likely to close and add blocks than nodes with a current stack of one. Thus, the more a node stacks, the higher the closing probability. This results in a model where nodes are encouraged to stack up their coins, rather than spend them.

New consensus models have appeared over time, proposing new incentive models and different objectives. For instance, the Proof of Activity (PoA) by Bentov et al. [1] is a merging concept of both PoW and PoS with a redistribution of the gathered fees to a set of randomly selected stakeholders. In the PoA consensus, miners use their computing power trying to add blocks to the current chain state. However, the miner solving the PoW will not earn the full pool of fees. A fixed number $n$ of stakeholders are sorted out with *follow the satoshi* (FTS)[2] algorithm using the hash of the previous block concatenated with $n$ fixed suffix values as input. Each selected stakeholder signs the block in a given time frame, one after the other, to prove their activity. The $n$ selected stakeholder are registered within the block. The fees assimilated as rewards are redistributed between the miner and the $n$ selected stakeholders. PoA incentives users to be connected to sign the new blocks and rewards them for it.

An improvement of the PoA model could be around blockchain usage, encouraging spending over stacking, keeping in mind that it is a fundamental criterion to sustaining the currency overtime. The Proof of Importance (PoI) aims to reward usage over stacking.

PoI is a consensus algorithm first introduced by NEM blockchain [4] and based on PoS and a scoring system. To be eligible to close a block, a node needs to have in its stack 10 000 coins. Each node has a score that increases according to the number of coins in its stack and the number of transactions initiated within the last 30 days. The larger and more frequent transactions, the greater the impact on the proof of importance score. Thus, the closing node will be selected according to its score, which will be reset upon block submission.

[1]Asymmetric mathematical problem is a mathematical problem that is hard to solve but where the solution is easy to verify.

[2]FTS is an algorithm that verifiably picks a coin $c$ and selects the owner of $c$ as a leader. Obviously the more coins the user has, the more likely they will be selected.

PoI is a consensus algorithm which encourages nodes to exchange their coins based on public infrastructure (i.e. a public blockchain), where everyone can access and read the ledger and write on it. Public blockchains are decentralized, meaning no single person has control over the network. Some examples of public blockchains are Bitcoin [13] and Ethereum [2]. Usually in public blockchains users are pseudonymized which does not necessarily mean anonymity. Indeed public blockchains record transactions in a fully shared decentralized ledger which by definition cannot offer privacy [11, 15].

Private blockchains (or permissioned blockchains) work similarly but with access controls that restrict those that can join the network. Private Blockchains have one or multiple entities who control the network, like third parties who regulate user transactions. A well-known example of private blockchain is Hyperledger [3].

In this paper, we propose the Proof of Usage (PoU) as a consensus that encourages nodes assimilated as users or participants to spend coins for service delivery. By using their coins, they sustain usage for a stable currency over time in an environment that is both permissioned and distributed. This consensus is introduced in the scope of the PikcioChain [5, 9]. The paper is organized as follows. The next section is devoted to our motivations and reasons to use a permissioned blockchain. Section III describes the protocol of our consensus and Section IV discusses different approaches to reward systems. The security of the PoU is presented in Section V and after which we conclude with a summary of what we have accomplished and a discussion of future research directions.

## II. MOTIVATIONS

Personal and identity data are the backbone of any service delivery, especially in the digital sphere. Anytime a customer wants to register for a loan (Fig. 1), the bank will require a complete identity evaluation through their data (identity card, name, age, address...). The data will be processed, crossed and finally validated to establish the customer's digital identity. Unfortunately, when this same customer is going to apply for insurance cover, they will be required to provide all this information again. As a result, both the bank and the insurance provider will check the data disjointedly before validating the customer's identity. This process is both time-consuming and costly with no particular return of investment for the company, and only leads to poor customer experience.

At the same time, data is a big market and many actors such as service providers buy data on the fly trying to enroll new consumers. Generally, this data is bought from third parties like social networks or email providers without the control or consent from its rightful owner i.e. the user (Fig. 2). In this model, the user is completely omitted from the business model despite being the sole provider of the data. To enable a more user-centric model encompassing control over their data, 2 directives can be proposed:

1) Reduce data prices from social networks,

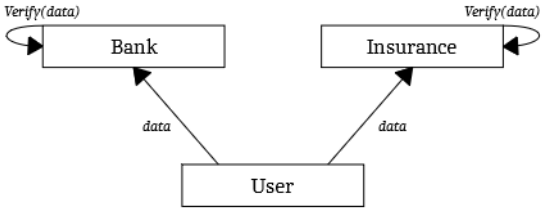


Fig. 1. Actual model of data exchange between bank, insurance and user.

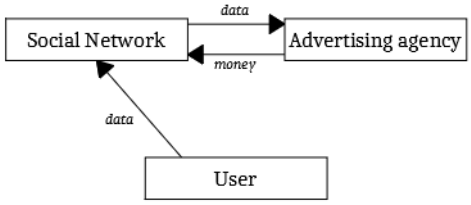


Fig. 2. Actual model of data exchange between social network, advertising agency and user.

2) or increase the price payed by advertising agencies.

The cases where 1) social networks reduce their profit and/or 2) advertising agencies increase their expenses can be discussed as a utopian model. If such a model can be proven to be both a carrier of more qualitative data and user-controlled in accordance with current regulations such as GDPR (General Data Protection Regulation [14]), it might enable a more virtuous data exchange process.

Our vision is to merge and empower (Fig. 1 and Fig. 2) in pursuit of a new model of data exchange between several entities (e.g. banks, insurance providers...) in a user-centric model (Fig. 3). In this model, users can provide their data to any entity they wish to register with, like a bank. As always, the bank verifies the customer's digital identity by processing their data. But this time, the user can allow an other party (e.g. insurance broker) to request and obtain their digital identity from the bank. The bank is both the personal identity carrier and the warrant of its validity. Such a quick enrollment process from the insurance broker is a strong benefit toward customer churn, where the bank can rightfully charge for the service. Such processes can be fully automated and decentralized in a consensus protocol. Hence, the Proof of Usage is introduced to both incentive usage and create new benefits for the data owner (i.e. private individuals) from business they generate with their personal data.

Such a model sustains each party's current income and includes a reward program for users with absolute control over their data.

Note that this model can only be viable in a permissioned blockchain, taking into account the holding of sensitive information within the DLT (Distributed Ledger Technology). GDPR aims for better and more virtuous data governance with which public blockchains cannot comply (see the General Data Protection Regulation [14] for more details).

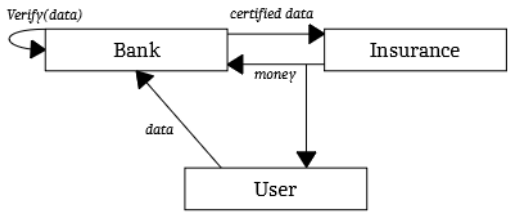


Fig. 3. Our model of data exchange between bank, insurance and user. “certified data" is data of user verified by bank.

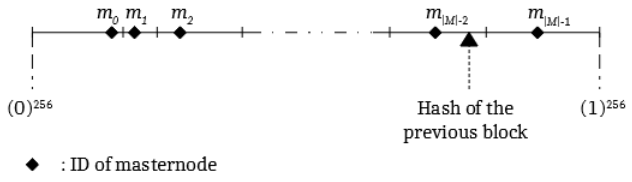


Fig. 4. Masternode selection system. In this example, $m_{|M|-2}$ is selected to seal and add the current block. $(0)^{256}$ and $(1)^{256}$ respectively correspond to the lower and higher bounds that a SHA-256 hash can take.

## III. PROTOCOL

PoU (Proof of Usage) can be perceived as an extension of the work from Bentov et al. on PoA [1] where users are incentive to exchange rather than simply being connected. This section describes the PoU process flow. PoU is based on the PoS selection principle. However, the scope does not apply to the global supply and dispatch of coins over the stakeholders but rather to the overall amount of exchange made within the last block of the current chain.

Note that cryptographic techniques based on an elliptic curve [12] are used to prevent spoofing and replays to detect corrupted transactions. Each contained transaction is secured using the public key of masternodes and signed with the sender’s private key. Transaction authentication codes and transaction digests are produced by collision-resistant hash functions.

In a permisioned blockchain, one cannot join the network without granted access by administrators. Participant and validator access is therefore restricted and controlled. Validators, called Masternodes, are nodes that verify transactions and add blocks to the chain. Consider a set $M$ of masternodes (numbered by $m_0, \ldots, m_{|M|-1}$) as validators and a set $U$ of user nodes. Only masternodes share the ledger and therefore are the sole entities with writing rights. Additionally, connections between masternodes form a complete graph i.e. each masternode is connected to all others.

Blockchain is a state transition system where a state consists of the ownership status of all existing tokens / coins and a state transition function that inputs a state and a set of transactions and outputs a new state. Let $S$ be a state of the blockchain and $changeState(S, T) = S'$ a state transition function inputs $S$ with a set of transactions $T$ (numbered by $t_0, \ldots, t_{|T|-1}$) and outputs as the new state $S'$. For example $changeState(\{A$: 25; $B$: 10 tokens$\}$,$\{A$ send 20 tokens to $B\}) = \{A$: 5 tokens; $B$: 30 tokens$\}$

In PoU we define the $changeState$ function as follows:

- select $m_s \in M$, the selected masternode adding a block $B$,
- $m_s$ select some transactions in its transaction stack to build up a set of validated transactions $T_B$,
- $m_s$ select “lucky nodes” that will earn the fees from the current block,
- $m_s$ creates the block, adds it to the blockchain to finally broadcast it over the network.

Hereafter, we develop the protocol phases.

### A. Masternode selection

The masternode selection process consists in selecting the masternode $m_s$ ($\in M$) which will close, sign and add a block to the blockchain. A unique ID is defined for each node and each masternode with a corresponding hash of a verified email concatenated with a unique username i.e. related to the node operator identity and a system salt. To select $m_s$, the hash of the previous block is compared to the set of masternode IDs (Fig. 4). The masternode which has the closest ID to the hash of the previous block is selected to be the next validator $m_s$ and therefore seals and pushes the current block. Note that all masternodes select the same $m_s$ without communicate because each of them knows the hash of the previous block and the set of masternode IDs. If a masternode is not available for any reason (disconnected, network latency, system failure...), protocol does not consider it and selects the nearest masternode available.

As observed in Fig. 4 all masternodes do not have the same probability of being selected even if the hash function is evenly distributed. For example, here $m_1$ is less likely than $m_0$ and $m_2$ to be selected. This cannot be perceived as an issue since the masternode property is auctioned. Therefore, masternodes which have higher probability might be more expensive to get than masternodes with a lower selection probability. The auction funds are redistributed to all the nodes equally.

### B. Transaction selection

In this paper we consider PoU as providing two types of transactions: *transaction* (note *Tx*) and *data transaction* (note *Tx-data*) (Fig. 5). *Transaction* is a classic transaction of coins from any node $A$ to another $B$. *Data transaction* is a transaction of data monetized using the system coins e.g. business $A$ pays in coins for certified user’s data $C$ from another business $B$ upon user’s consent. When a node initializes a transaction, it sends its request to all masternodes. *Data transaction* is initialized, pending until the node which pays coins receives certified data. This pending state prevents malicious nodes from trying to get paid without sending the data.

When masternodes receive a request, they make sure the transaction is valid by verifying the node signature and the current balance in hold. If validated, the transaction is added

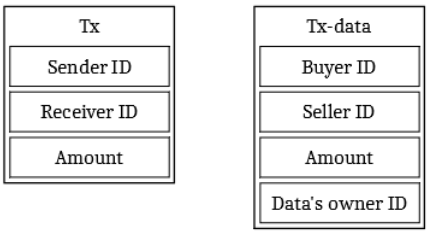


Fig. 5. Transactions model.

| Transactions | | | | |
|---|---|---|---|---|
| Sender /buyer | Transactions type | Receiver /seller | Amount in coin | Data's owner |
| A | Tx | C | 15 | - |
| B | Tx-data | D | 13 | E |
| C | Tx | B | 4 | - |
| A | Tx | B | 10 | - |
| D | Tx-data | F | 8 | C |

Fig. 6. Example of set of transactions of a block.

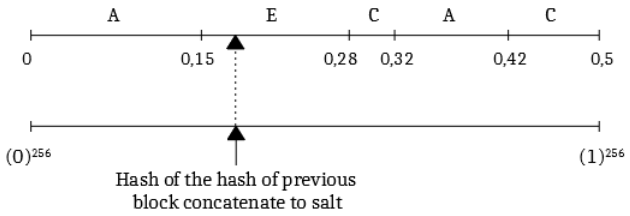


Fig. 7. Example of a node selection with the FTS algorithm. Here, a node $E$ owns the coin determined by the hash of the hash of previous block concatenate to a salt, then $E$ earn the transactions fees.

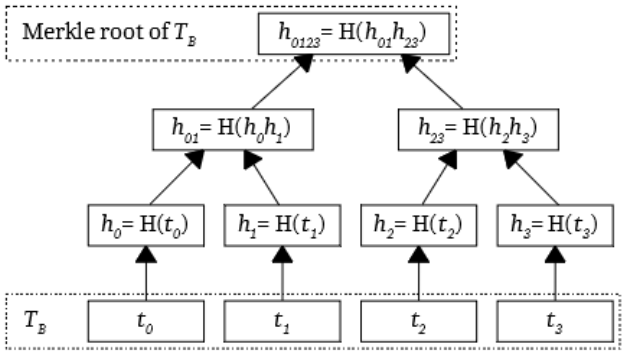


Fig. 8. Construction of the Merkle root of $T_B$, function H() corresponding to any hash function.

to the current transaction stack to await being sealed into the block.

When a masternode $m_s$ is selected to create a block $B$, like any other it stacks up transactions in the $T_B$ set for the current block (see example in Fig. 6). For technical reasons, the number of transactions in a block is bounded.

### C. "Lucky nodes" selection

The "lucky nodes" are the $n$ nodes which share the rewards of the block with the selected masternode $m_s$ (report to Section IV for more details). Selection of the $n$ lucky nodes is based on Follow The Satoshi (FTS) algorithm. To use the FTS algorithm, consider that each coin of the network is signed and traceable. The FTS algorithm in the scope of PoS is used to select $n$ nodes, by selecting $n$ coins in the network supply. Owners of the selected coins are the lucky nodes. However, in the case of PoU, the hash of the previous block is concatenated with $n$ secret salts (known by all masternodes) to select $n$ nodes from the current transaction set $T_B$ of the current block $B$.

The number $n$ of nodes to select can be discussed. In this particular case, PoU can select a fixed number of nodes (one or more) or a number scaled according to the block's number of transactions or the total amount of exchanged coin in it.

Let's consider the previous example given by Fig. 6 and a transaction fees rate of $1\%$ of the total amount in transaction stack $T_B$. The FTS algorithm is executed as displayed on Fig. 7. In this particular case, a node $E$ is selected and gets the transaction fees as a reward for using the service.

### D. Share the block

At this stage, $m_s$ has validated transactions in its stack, to create the $T_B$ set to include in block $B$. Now $m_s$ builds up the header of the block, containing the hash of the previous block i.e *Hash*($B-1$), the Merkle root [10] of $T_B$ (Fig. 8), the ID list of lucky nodes selected, and time stamp of $B$. Finally $m_s$ signs the header with its private key to lock and secure the header. Note that the hash of a $B$ block corresponds to the hash of $T_B$ concatenated with the hash of the previous block $B-1$.

At this point, $m_s$ has successfully sealed the $B$ block and added it to its chain (Fig. 9). Then, $m_s$ broadcasts the new block to all other masternodes. When a masternode receives a block from another, it validates the header and content to finally add it to its chain. A receiving masternode $m_r$ considers a block $B$ as valid if:

1) previous hash in the $B$ block corresponds to the hash from the last block of the current chain of $m_r$,
2) $m_s$ selected masternode ID is indeed the closest ID from the previous block hash (i.e $m_s$ is the rightful validator),
3) all transactions within $B$ (i.e all transactions of $T_B$) are valid (signature and sufficient funds),
4) lucky nodes selection is not biased (e.g $m_s$ rewards the rightful nodes).

After receiving a valid block, $m_r$ adds it to its chain and deletes processed transactions from its transaction stack. Condition 1) cannot be filled in two following cases: (i) $m_s$ tries to share a block with false information or (ii) $m_r$ is not up-to-date and its last block is late from the current chain of $m_s$. Case (ii) can happen when $m_r$ did not receive previous block(s) or when it reconnects to the network after some inactivity or failure. When 1) is not complied with, $m-r$ proceed to a recovery protocol. Let's consider $h$ the hash of

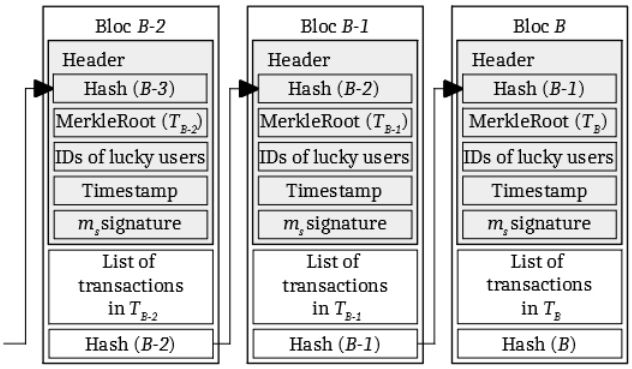


Fig. 9. Structure of the blockchain.

the last block $B$ of $m_r$. The recovery protocol proceeds as follows:

1) $m_r$ requests the hash of the current block to all masternodes.
2) Masternodes send the hash of their current block to $m_r$.
3) $m_r$ selects the hash with the most occurrences $h'$ and compares it with $h$. If $h = h'$ the masternode in recovery is up-to-date and the protocol is terminated (corresponding to case (i)). Otherwise it continues, following the next steps.
4) $m_r$ requests all the missing blocks starting from its $B$ to any masternode with rightful last sealed block.
5) $m_r$ validates the block depending on its updating flow as described in section D.

This protocol allows $m_r$ to rebuild the missing parts of the ledger from its last known block and the chain last block.

### E. Execution of the changeState function

The changeState function is executed at regular intervals e.g. 20 seconds. If the selected masternode $m_s$ does not share a valid block after a fixed number of executions (e.g. 2 block time), the function will consider it unavailable. Therefore the block closing the masternode process will omit the ID of the unavailable masternode. This condition prevents the network from inflating in a pending state where $m_s$ is disconnected in a failure status to process normally.

## IV. APPROACHES

This section explores the reward / incentive model, i.e how to incentive masternodes processing transactions and blocks, and reward lucky nodes selected within the current block. Rewards are issued directly from the selected masternode to the lucky nodes.

### A. Model with transaction fees

Common reward systems rely on transaction fees. When a node spends coins for a given transaction, a percentage $\alpha$ of the transaction amount is extracted as a reward. Thus, every time the selected masternode $m_s$ adds a block, it earns a percentage $\beta$ of the total reward within the current block and redistributes 100 - $\beta$ to the lucky nodes. Such a system usually works with a deflationary currency but it can be adapted to an inflationary currency by generating / minting $c$ new coins as reward. Note that $c$ is not necessarily a constant, it can also be determined as a percentage of the global reward.

The main problem with this transaction fee-based system is for "classic nodes" and private individuals who are not necessarily familiar with such models and cryptocurrency ecosystems in general. In the common fiduciary bank model, when a user spends 10 coins, the receiver receives the whole value and therefore does not expect to pay additional fees for this service.

### B. Model without transaction fees

Another reward system is therefore to create the reward. In this system, fees are calculated as described in the previous approach as $\alpha$ percentage of the transaction amount. Note that such a system is inevitably inflationary but when a node $A$ exchanges 10 coins to another $B$, $A$ pays 10 coins and $B$ receives 10 coins.

However, this approach cannot prevent nodes from spamming huge amounts of transactions across multiple accounts. Hence, such malicious nodes won't spend any related fees which could break the attempt to participate in the lottery during each block's sealing process.

A solution to this is to limit transactions for each node over time, like any conventional banking system. This limit can be defined according to the number of transactions and the total amount allowed for free within a given time frame. If a node spends more than the authorized limit, transaction fees will be withdrawn accordingly. This limit prevents malicious nodes because they will start to spend fees that they will potentially never retrieve.

From our point of view, minting new coins based on a transaction fee system with limit control is an approach that can enhance usage and understanding from private individuals, i.e. users. This system is a balanced compromise between "common users" who do not want to pay fees regardless of their scarce usage and nodes trying to pervert the system.

## V. SECURITY

This section presents how the PoU prevents attacks from malicious nodes in order to build a resilient and reliable consensus. We will not consider attacks from masternodes because they are normally trusted. The PoU is designed for permissioned blockchains with a strong user identification to empower trust. Identification is a strong vector of trust and easy to set up in decentralized environments. As a result, a given entity cannot create plural identities.

### Denial of service attacks

Denial of service (DoS) attacks or distributed denial of service (DDoS) attacks in the PoU imply the possibility for a node to spam the network. But as discussed in Section IV a node will be deterrent to act due to the transaction fees withdrawn accordingly. If one or several nodes try to

spam the network they will quickly reach their limit and pay transaction fees that won't be worth their gain.

*Double spending*

Considering 51 percent of masternodes as honest, when a transaction is stacked in $|M|/2$ chains of masternodes, the transaction is considered fully validated. This is enforced by the recovery protocol, where $|M|/2$ masternodes hold a block containing a given transaction. All others will eventually hold it as well if they want to continue supporting the PoU and its reward process. If a node wants to double spend, it has to be sure that at least $|M|/2$ masternodes have registered its transaction.

*Power of users pool*

Generally, distributed systems suffer from groups of users (called "pool of users") who join forces to increase their power over the blockchain depending of course on the type of consensus in place. For example, in the PoW the power is the computing capacity and in the PoS it depends on the number of coins in user's balance. In the PoU, users who have the power are users who exchange coins indirectly on the number of coins they own. This means users without coins cannot exchange them and the chances of winning increase with the amount of the transaction. Consider a group of users who create a pool. To increase their chances of being selected as lucky nodes, users need to increase the amount of their transactions. Thus, users in the pool need to stack up coins on a given account and move it from one to another to generate big transactions without exceeding the transaction limit before paying fees. Now let's look at what happens if a pool user is malicious towards other participants. Malicious users can wait for the moment when they have all the pool coins on their public key and therefore retain them. Users of the pool cannot prevent this kind of behavior, which creates a high-risk environment. Pools of users are still possible, but trust is the essential vector to render them viable. Smart contracts can interplay this movement over time to secure such pools. Pools of users will certainly be created as part of PoU, the transaction limit will restrain their capacity on the network.

*Fault tolerance from masternodes*

Let's consider faulty masternodes that can disconnect or suffer from latency trying to keep the blockchain up-to-date. While at least a relative majority of them agree on the last block, the system is safe. This is implied by the recovery protocol, with at least a relative majority having the same last block. When back on the network, faulty masternodes can retrieve all missing blocks from any proven up-to-date masternode from the majority pool.

## VI. CONCLUSIONS

This paper describes a new consensus algorithm that encourages users to spend coins for services rather than stacking, to promote and sustain usage for a stable currency over time in an environment that is both permissioned and distributed.

From our point of view, the model described in Section II is a great alternative to current data exchange models, supported by Proof of Usage.

Proof of Usage is already up and running on the PikcioChain testnet and is undergoing extensive testing and benchmarking. Proof of Usage in its first version is open sourced at *https://github.com/Pikciochain/Proof_of_Usage*. It shows promising results and carries new usage every day. Future works will consist in pushing over security measures to evaluate scalability metrics such as the processing number of transactions in a given time frame.

## REFERENCES


[1] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. proof of activity: Extending bitcoin's proof of work via proof of stake. *ACM SIGMETRICS Performance Evaluation Review*, 2014.

[2] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[3] Christian Cachin. Architecture of the hyperledger blockchain fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, volume 310, 2016.

[4] NEM company. Nem, technical reference, version 1.2.1. 2018. *https://nem.io/wp-content/themes/nem/files/NEM_techRef.pdf*.

[5] Pikcio Company. Pikciochain, the personal data chain, white paper version 2.0. 2018. *https://www.pikcio.com/pikciochain-whitepaper*.

[6] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Annual International Cryptology Conference*, pages 139–147. Springer, 1992.

[7] Markus Jakobsson and Ari Juels. Proofs of work and bread pudding protocols. In *Secure Information Networks*, pages 258–272. Springer, 1999.

[8] Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper*, 2012.

[9] Jorick Lartigau, Fabien Bucamp, and Didier Collin de Casaubon. Pikciochain: a new eco-system for personal data.

[10] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.

[11] Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 397–411. IEEE, 2013.

[12] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

[13] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[14] General Data Protection Regulation. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46. *Official Journal of the European Union (OJ)*, 2016. *https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L:2016:119:FULL*.

[15] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy (SP)*, pages 459–474. IEEE, 2014.

[16] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. An overview of blockchain technology: Architecture, consensus, and future trends. In *Big Data (BigData Congress), 2017 IEEE International Congress on*, pages 557–564. IEEE, 2017.