Islamic University of Technology (IUT)
Department of Computer Science and Engineering (CSE)

PPoS: An Optimized Consensus Protocol for IoT Devices

Authors

Tasnim Ferdous Anan - 180041108

Abdullah Ibne Masud Mahi - 180041114

Tausif Khan Arnob - 180041138

Supervisor

Dr. Md. Sakhawat Hossen

Associate Professor,

Department of CSE,

Islamic University of Technology (IUT)

Co-Supervisor

Faisal Hussain

Assistant Professor,

Department of CSE,

Islamic University of Technology (IUT)

A thesis submitted to the Department of CSE in partial fulfillment of the requirements for the degree of B.Sc. Engineering in CSE

Academic Year: 2021-22

May - 2023

Declaration of Authorship

This is to certify that the work presented in this thesis is the result of research and experiments conducted by **Tasnim Ferdous Anan**, **Abdullah Ibne Masud Mahi**, and **Tausif Khan Arnob** under the supervision of **Dr. Md. Sakhawat Hossen**, Associate Professor, Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh. It is also stated that neither this thesis nor any portion of this thesis has been submitted for any other degree or diploma. The book acknowledges information obtained from the published and unpublished work of others and includes a list of references.

Authors:

Tasnim Ferdous Anan

Student ID - 180041108

Abdullah Ibne Masud Mahi

Student ID - 180041114

Tausif Khan Arnob

Student ID - 180041138

Co-supervisor:

Faisal Hussain

Assistant Professor

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

Supervisor:

Dr. Md. Sakhawat Hossen

Associate Professor

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

Acknowledgement

We would like to thank **Dr. Md. Sakhawat Hossen**, Associate Professor, Department of Computer Science Engineering, IUT, for serving as our adviser and mentor. His enthusiasm, thoughts, and insights have been vital to this research. This research would not have been possible without his help and guidance. His valuable opinion, time, and input were provided throughout the thesis work, from the first phase of thesis topic introduction, subject selection, algorithm proposal, modification, and project implementation and finalization, which assisted us in doing our thesis work properly. We are really grateful to him.

We are also grateful to **Dr. Md Moniruzzaman**, Assistant Professor, Department of Computer Science & Engineering, IUT and **Faisal Hussain**, Assistant Professor, Department of Computer Science & Engineering, IUT for their valuable inspection and suggestions.

Abstract

Internet of Things (IoT) devices are currently on the rise, with homes and businesses constantly adding and replacing old mechanisms with new, smart IoT devices. However, security vulnerabilities are also becoming more apparent as a result and the use of Blockchains can help reduce some aspects of these vulnerabilities. Most of these blockchains are based on private, permissioned architectures, which while are perfect for private solutions, present a new set of problems when applied to public infrastructures. We thus tried to focus on the applications of public blockchains on IoT devices.

TinyEVM presents a novel solution to this problem. However TinyEVM is optimized mostly for one-to-one communication, thus having inherent scalability issues. Current consensus protocols for side-chains can be used to solve this problem, but they are either not optimal or have security risks. In this paper we thus propose a novel consensus protocol for side-chains, tailored for use in IoT devices in conjunction with TinyEVM which would enable the use of the public blockchain, Ethereum on most IoT devices. Periodic Proof of Stake (PPoS) is a consensus protocol based on Delegated Proof of Stake (DPoS) but with a Trust model and a focus on minimizing network traffic through performing consensus periodically and multi-casting transactions to selected nodes instead of whispering. PPoS also provides flexibility to choose between security and energy-efficiency. Traffic, performance and security analysis shows the improvements compared to existing consensus protocols.

Keywords- Blockchain, Internet of Things, Consensus, Ethereum, Side-chain

Contents

1 Introduction

With the increasing spread of smart devices and the production of highly capable low power IoT[17] devices, modern businesses and industries have taken to automating as many processes as possible. This has led to the permeation of these battery-powered capable IoT devices into almost all types mechanisms that deal with monetary transactions. As a result, we often see the general populace carrying only their smartphones as their primary form of payment, through either taps or through scanning QR codes.

What we must remember, though, is that all of these multitude of transactions are carried out through third parties, this thus face all the typical problems associated with them such as no transparency, intermediate charges and long wait times in case of any response to mishaps, if any. Blockchains, on the other hand, are immutable, public and distributed ledgers that enforce fairness and transparency. As a result, two untrusted parties can easily carry out multiple transactions using blockchains without any of the demerits associated with third parties.

TinyEVM[3] solves many of the problems that previously hindered the use of public blockchains in low-powered IoT devices, however it specializes in transaction between only two parties.Therefore, considering businesses like theme parks and game centres that rely on multiple micro-transactions across a large number of attractions, TinyEVM fails to deliver. A possible solution is to modify TinyEVM to accommodate an existing consensus protocol on it's side-chain layer to enable multi-party transaction. Existing solutions, however, rely heavily on broadcasting and transaction pools to ensure fairness and security, which makes them highly energy-inefficient.

To overcome both of these challenges, we propose Periodic Proof of Stake (PPoS), a novel consensus algorithm that utilizes the concept of delaying transactions and multi-casting to optimize energy efficiency. Firstly, we let the nodes accumulate transactions before sending them in one go to minimize traffic congestion, collisions and excessive overheads. Secondly, we have the nodes multi-cast packets to relevant parties instead of broadcasting to the whole network to again reduce collisions and improve efficiency. Our goal is to introduce a flexible, secure and energy efficient algorithm that compensates for the deficiencies in TinyEVM while also being applicable in other similar scenarios.

To summarize, our contributions are as follows:

- We address the lack of suitable consensus protocols for public blockchains in IoT devices.
- We propose a tentative consensus protocol for multi-party public blockchains in IoT devices.
- We provide a mechanism that how our proposed protocol can minimize network traffic during consensus period and thus improving efficiency compared to existing systems.

2 Background Studies

2.1 Blockchain

2.1.1 Overview

A blockchain is quite literally a chain of blocks. Each block acts as a ledger containing a number of transactions and their details. Blocks in a blockchain contain a reference to the previous block in the chain, hence having all the blocks connected like a chain. Multiple peers can participate in a blockchain network and each peer has a full copy of the blockchain. As a node can connect to a blockchain network from anywhere as a trusted or untrusted party, a blockchain network enables decentralization as the transactions are transparent to anyone participating in the network or to a node with appropriate permissions. Due to this decentralized nature, the ledgers in a blockchain can be said to be immutable enabling high security and transparency.

Due to the vast number of users currently participating in the internet, the greatest problem faced by businesses providing services is security. Most of these businesses are centralized and the data they store is too great for decentralized redundancy. As a result, malicious attackers can directly attack these services or data storages and steal or manipulate data. The most common types of attacks used nowadays are DDoS, ARP spoofing, data manipulation and network congestion. The common suit amongst all these attacks are that they are only successful if the services provided are centralized. Blockchain by its very nature is decentralized. Attackers would need to successfully deprive multiple peers in order to prevent users from accessing these services, and the greater the decentralization, the greater the computing power required for these attacks. Another point to consider is that as the amount of computing power an average user has access to increases with time, companies have to invest more in encryption and cryptography to resist these attacks. This leads to higher computational requirements as well as large bandwidth requirements, which impose highest costs on businesses. Since data stored in a blockchain is immutable, aside from the problem of data confidentiality, blockchain is almost an ideal solution to most forms of cyber-attacks, with minimal costs and overhead.

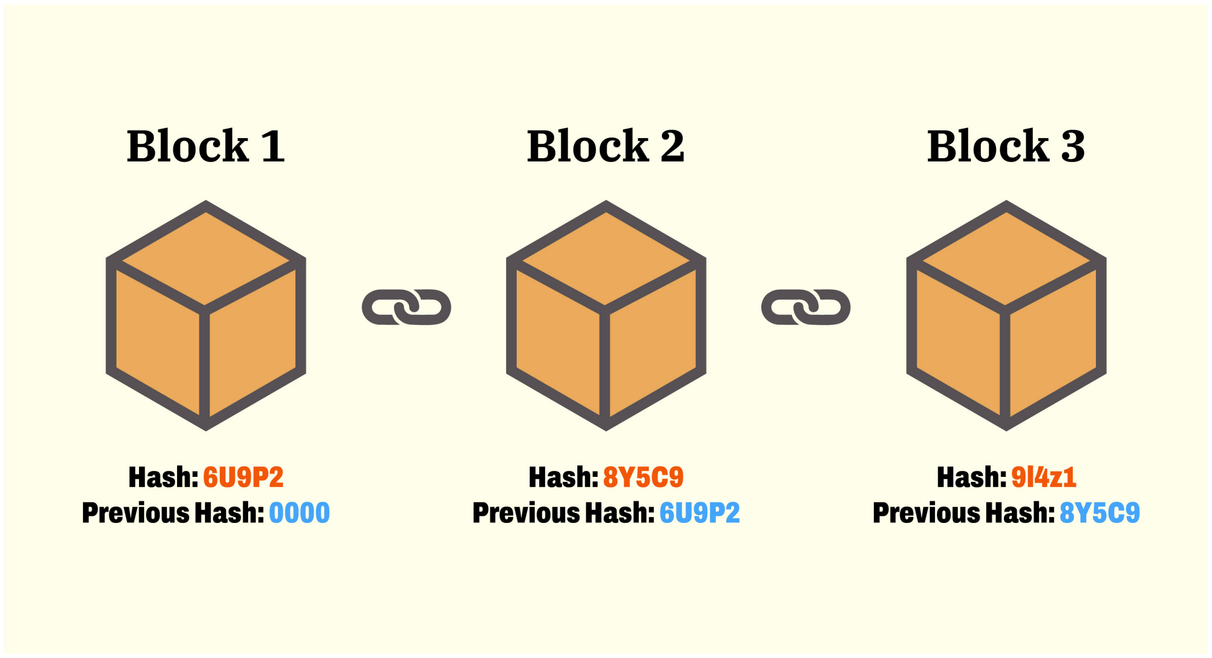


Figure 1: Blockchain Infographic

Blockchains can be classified into two categories: permissionless and permissioned blockchains. In permissioned blockchains, only authorized nodes can be added and act as peers. There is usually an authority that assigns certificates to participants to verify their identities. In permissionless blockchains, any node on the internet can participate and act as a peer without needing to have their identity verified. Different blockchain architectures have been proposed over the years, but three particular systems stand out: Bitcoin, Ethereum and Hyperledger.

2.1.2 Bitcoin[1]

In 2008, Satoshi Nakamoto became the first person to design and implement a complete blockchain network and named it Bitcoin[1]. It served as a form of cryptocurrency as well as a platform to perform transactions using said cryptocurrency. This was possible only by creating a peer-to-peer timestamp server that maintained the order and immutability of transactions by generating cryptographic proofs, allowing two or more unknown untrusted individuals to safely make transactions. There are a number of elements that make up the bitcoin architecture:

- *Digital Signatures*: Every participating node is issued a public and private key pair. The sender or the one making the transaction must sign or encrypt the data using their own private key. The resulting encrypted message can only be decrypted by the corresponding public key. There is no method to recalculate the private key and all public/private key pairs are unique. This digital signature is used to ensure that the person making the transaction is authentic and not an impersonator.

- *Transaction Pool*: Peers in the network broadcast transactions to all the other nodes and these transactions are stored in a transaction pool with a timestamp. Special nodes called Miners then choose transactions from this transaction pool to add them to a block.

- *Block*: A block in a bitcoin network is simply a kind of ledger. It contains a cryptographic hash value of the previous block, the merkle root of all the transactions added to the block and a nonce value calculated by miners.

- *Merkle Tree*: All the data to be added to a merkle tree are hashed and added to the leaf nodes. The hash of two adjacent leaf nodes are concatenated and hashed again forming intermediate nodes. This process is repeated, with single nodes being hashed with themselves, until only one node remains, which becomes the merkle root.

- *Consensus Protocol*: A consensus protocol is a set of rules or algorithms that are used to achieve agreement among the members of a distributed system. In a blockchain system, a consensus process is used to ensure that all participants agree on the status of the shared ledger and to prevent contradictory transactions from being recorded on the ledger.

  There are several forms of consensus protocols, including proof of work (PoW), proof of stake (PoS), delegated proof of stake (DPoS), and proof of elapsed time (PoET), among others. For validating transactions and adding new blocks to the blockchain, each type of consensus mechanism has its unique set of rules.

  Consensus protocols are an important part of blockchain technology because they allow for secure and decentralized decision-making in a distributed system. They help to ensure the integrity and immutability of the blockchain, and they enable trust to be established among parties who may not know each other or fully trust each other.

To create a block in a bitcoin network, miners collect transactions from the transaction pool and try to solve a cryptographic puzzle. This is the basis of the consensus mechanism in bitcoin and it is a Proof of Work (PoW) consensus. The puzzle is a race to calculate a nonce value, which yields a hash value of a block with a set number of leading zeros, the number of zeros signifying the difficulty of the puzzle. As a block contains the hash of the previous block and a merkle root, any change in the previous block would alter the hash value of the current block, which means the the current block and all subsequent blocks will need to have their hash values recalculated to maintain the cryptographic puzzle, which demand tremendous computation power.

Once a miner has solved the puzzle, they broadcast the block to all the other peers in the network who validate the block by checking the integrity of the transactions as well as the puzzle. Once a block is verified, it is added to the chain. It is to be noted that the first block to be verified is added to the chain, the rest are discarded.

A malicious attacker may try to add invalid transactions to a block or add an already existing transaction again, which is the problem of double spending. To do this, they would need more than 50% of the computation power of all the participating nodes. This is because during the consensus mechanism, peers do not add a block to their chain if the block is found to be invalid. As a result, if the attacker tries to add his own malicious block, they must create a fork in the chain. Bitcoin only accepts the longest chain and peers only add valid blocks to the chain. In the case of an invalid block, the attacker must keep adding new blocks faster than all the other peers, which means that they must have a computational power greater than all of the peers combined. With greater decentralization and more peers participating in a network, this becomes infeasible, hence the high security and immutability of blockchains and therefore bitcoin.

2.1.3 Ethereum[2]

Bitcoin allows the use of scripts in order to verify the public keys and transaction data, but it has several limitations. Firstly, the scripts are not turing complete, due to the absence of loops. Secondly, there is no way to implement complex conversion like BTC to USD and vice versa or to implement complex conditions before a transaction is carried out.

Ethereum expands on the concept of bitcoin by adding a Turing-complete programming language which allows anyone to create their own currency on top of ethereum or any type of decentralized application that any participating node can run, called smart contracts[2]. The currency of Ethereum is Ether and it is required to pay transaction fees. The smallest denomination of Ether is called GAS and every step of computation in a smart contract requires the payment of GAS. Every smart contract also has a GAS limit and both the price of GAS and GAS limit prevent misuse of computational power through the use of malicious code in smart contracts like infinite loops. Smart Contracts run on a virtual machine called the “Ethereum Virtual machine” or EVM. The EVM uses a stack, an infinitely expandable memory (byte-array) and a persistent contract storage.

Unlike bitcoin, the most recent state along with the transaction list are included in Ethereum blocks.

2.1.4 Hyperledger

Unlike Bitcoin and Ethereum, Hyperledger Fabric (Fabric) was designed to be permissioned, targeted mainly towards private businesses[18]. The consensus protocol can also be changed unlike the other two for more specialized use cases and the smart contract alternative supports the use of general programming languages like go, java, node.js etc unlike a domain specific language like solidity for ethereum. The execute-order-validate model also differentiates it from the order-execute-validate model used by the other two. Fabric also tries to solve the limitations of sequential execution of transactions by the execute-order model while allowing better confidentiality of smart contracts and eliminating non-deterministic transactions.

Every Fabric system must have and ordering service to order transactions, a membership service provider that validates and assigns identities to participating nodes, a peer-to-peer gossip service that broadcasts the orderer’s output, a sandbox environment to run chaincodes, a smart contract alternative and peers that maintain the ledger and participate in validating transactions.

Previous blockchain architectures followed the order-execute model. This means that once the block has been mined and added to the chain, all the nodes in the network must execute all of the transactions again for validation, resulting in wastage. The throughput is also restricted as all transactions have to be executed serially. Another issue is that due to the fault tolerance for the consensus protocol being around 1/3rd, the consensus protocol usually fails in small scale applications due to all participating nodes having the same authority, making it easier for a malicious attacker to introduce faulty blocks.

Fabric has three types of participants: clients who request transaction execution, endorsers that execute transactions and present them to the orderer and validators who check the orderer's output. Multiple blockchains, called channels, can be connected to a single Fabric network and they may have individual orderers or a single central one.

Following the execute-order-validate model, in the execution phase, endorsers execute and validate transactions proposed by clients who must sign it using their provided keys by the MSP. The endorsers run a simulation of the transaction in an isolated environment and send the output back to the client. The client must then collect a sufficient number of endorsements to satisfy the endorsement policy and then present it to the orderer. In the ordering phase, the orderer collects multiple transactions and arranges them into a single block that is totally ordered. In the validation phase, the orderer broadcasts the block to all the peers. The validators then check the endorsement policy and the read-write states and once they are confirmed to be valid, they update the ledger and current world state, which has remained unchanged until now, allowing other peers to follow through. The consensus protocol, peer gossip protocol, ordering protocol, endorsement policy, chaincode language are all flexible in hyperledger. Even peers can be separated into groups or organizations which trust themselves but not those outside their group.

2.1.5 Sidechain

A sidechain[19] is a separate blockchain that is connected to a main blockchain, allowing assets to be transferred between the two chains. Sidechains can be used to experiment with new features or to scale a blockchain by moving some transactions off the main chain and onto the sidechain.

One of the main benefits of sidechains is that they allow for the creation of new features or the testing of new technologies without affecting the security or stability of the main chain. They also enable the separation of different sorts of transactions, allowing the main chain to focus on its essential functions.

To transfer assets between the main chain and the sidechain, a procedure known as ”pegging” is used. This entails connecting the two chains in two directions, allowing

11 assets to be swapped back and forth. The specific technique for transferring assets between a main chain and a sidechain may differ depending on the implementation.

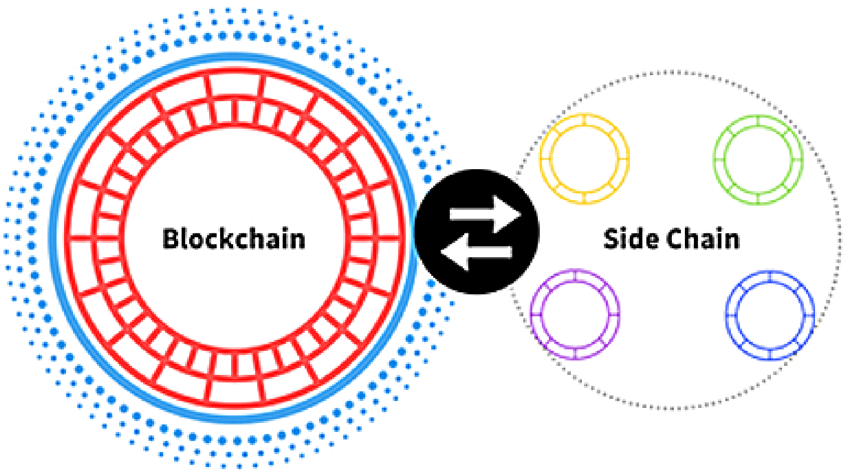


Figure 2: Sidechain Infographic

2.1.6 Off-chain

Off-chain[20] transactions happen outside of the main blockchain. They can be used to scale a blockchain by moving some transactions from the main chain to an off-chain environment where they can be processed more efficiently.

Off-chain transactions can be accomplished through payment methods. Payment channels enable many transactions to occur between two parties without requiring each transaction to be recorded on the blockchain. This can help to minimize the load on the main chain and enhance overall system efficiency.

Another way to conduct off-chain transactions is through the use of sidechains, which are separate blockchains that are connected to the main chain. Assets can be transferred between the main chain and the sidechain, allowing for some transactions to take place off the main chain.

Off-chain transactions can help to improve the scalability and efficiency of a blockchain system, but they also have some potential drawbacks, such as the need for additional trust between parties and the potential for reduced transparency.

2.2 Blockchain in IoT

As our daily lives are becoming increasingly dependent on technology, devices like sensors, cameras and actuators are becoming more and more important. These devices, henceforth referred to as IoT devices may serve a number of purposes such as data collection, environment monitoring, alarms and controlling other IoT devices[21]. These devices are not manufactured by any single company, nor do they produce data in the same formats. However, entities that must utilize these devices to provide services to end users must devise methods to communicate with and manage these IoT devices through complex hierarchical systems with a lot of computational and communication overheads. These overheads usually lead to major delays in potentially life-risking events such as in the case of fire alarms notifying the fire department, medical monitoring devices notifying doctors of patients in critical conditions and the like. Moreover, since different entities usually manage different devices, communication between two or more untrusted devices also requires additional overheads, such as an electric car negotiating with a parking meter. Furthermore, these problems add additional overheads, sometimes not even allowing communication between similar devices, they also pose many security risks due to multiple complex implementations. Blockchain provides a solution to most of these problems[22][23][24]. Not only is the data stored in a blockchain ledger immutable, pieces of code that are transparent and immutable can also be run on the blockchain network to facilitate intercommunication between trusted and untrusted devices, even with devices producing data in different formats. Additionally, this decentralized software and data are safe from malicious attacks owing to the nature of blockchains.

2.3 Limitations of Blockchain in IoT

The peers in a blockchain node usually have one or more responsibilities, most of which are computationally demanding or energy-consuming. IoT devices are designed to be lightweight and energy efficient to keep costs low and ensure self-sufficiency in terms of energy for long periods. As a result, the common blockchain technologies cannot be directly implemented in these IoT devices and require adjustments to address the different issues associated with these devices[25][26].

- **Energy Supply** - Most IoT devices run on batteries that may be periodically recharged or attached to self-sufficient power sources like solar panels. As a result, the amount of energy they can store or use is very small and limited. For a blockchain network to work properly, constant high communication bandwidth is required which means that these IoT devices, which are mostly wireless, need to consume huge amounts of energy to transmit and receive data; energy that is not available to them.

- **Computation Power** - The core security aspect of a blockchain network is maintained by its consensus protocol and multiple validations. However, these operations are very computationally expensive for IoT devices which use processors that operate with clock speeds in the MHz range. Due to the limited computational power available to them, joining a blockchain network as a peer is implausible.

- **Memory** - Along with the low computational power, low memory capacities ranging from a few dozen kilobytes to a few megabytes is very commonly seen in IoT devices. A bitcoin block is at a minimum of 1 megabyte in size, hugely outclassing the memory capacity of most IoT devices, let alone leaving space for computations.

- **Throughput** - Some IoT devices like medical monitoring devices produce a constant stream of data in large amounts. Current popularized blockchain architectures have a limit on the throughput of transactions, preventing large-scale implementations in the case of such IoT devices.

3 Related Works

Throughout the years, a number of solutions pertaining to the limitation of implementing blockchain networks using lightweight IoT devices have been proposed[27][28][29][30][31]. All of these solutions deal with one or more aspects but provide no overall solution. In this section, we will explore some of the more comprehensive solutions proposed concisely.

3.1 Existing Solutions to Limitations of Blockchain in IoT

3.1.1 TinyEVM: Off-Chain Smart Contracts on Low-Power IoT Devices[3]

Bitcoin and Ethereum are the most popular architectures for making cryptocurrency transactions. However, the PoW consensus algorithm used is very resource-intensive. The authors in this paper have thus proposed a solution to this problem using Ethereum[3].To ensure the validity of the data collected, data generation by the sensors in IoT devices should best be handled by smart contracts, which are transparent and immutable. Nevertheless, Ethereum has no solution to handle sensors using smart contracts. Therefore, the first contribution that the authors made was to modify some of the existing opcodes in the Ethereum Virtual Machine (EVM)[32] to allow for smart contracts to directly handle sensors.

Along with the modification of the underlying opcodes, the authors have also proposed a solution to the resource-intensive PoW consensus that needs to be handled by any participating node. The solution was to create a separate side-chain branching off from the main ethereum network. The concept works in three phases:

- A smart contract is deployed in the main chain that enables the construction as well as the handling of an off-chain (side-chain) branch. The smart contract runs on the main chain and a specific amount of funds are locked for use in the off-chain to handle transactions.

- The node responsible for creating the off-chain deploy another smart contract that deals with all of the data generated by the IoT devices connected to that node. All the transactions generated by the IoT devices are collected, validated and kept offline until the node communicates with the main chain again, which is done periodically. A sequence counter is maintained by each device for all of the transactions performed during this period to avoid synchronization issues and to keep track of the order of transactions.

- Once a node reconnects to the main chain, the final amount of the transactions are then pushed to the main chain transaction pool to be added to a block, validated and then committed to the main chain. The total amount of currency exchanged is limited by the initial amount of funds locked to the off-chain.

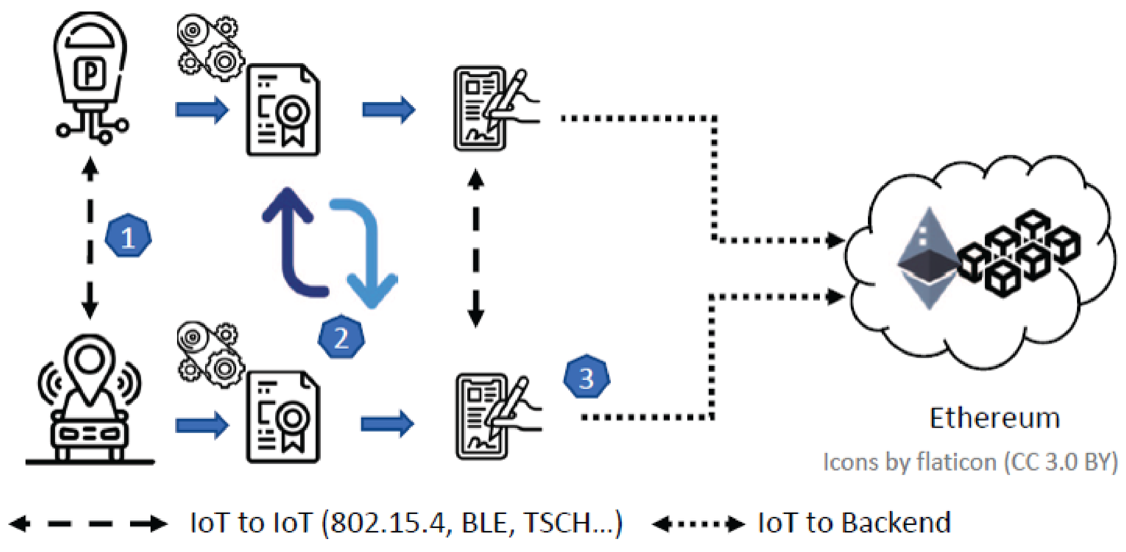

1
2
3
Ethereum
Icons by flaticon (CC 3.0 BY)
IoT to IoT (802.15.4, BLE, TSCH...)
IoT to Backend

Figure 3: TinyEVM

3.1.2 PoBT: A lightweight consensus algorithm for scalable IoT business blockchain[4]

The network architecture proposed in this paper[4] is the same as that of the Hyperledger Fabric but with one additional detail; the IoT devices are not directly connected to the network as peers. Instead, they act as clients that give their request to peer nodes. Each peer node performs the same duties as a normal Hyperledger Fabric peer node, but serves multiple IoT devices to whom they are connected, with the connections being called channels. IoT devices can only communicate with each other with the node or nodes as intermediaries. As a result, if devices on the same node want to communicate with each other, a local consensus is carried out saving bandwidth and thus increasing scalability[33][34].

3.1.3 Towards an optimized blockchain for IoT[5]

A novel idea was put forth in this paper[5] in the form of creating a lightweight architecture. We can see some similarities with the hyperledger framework with nodes being given different responsibilities, but that is mostly where the similarities end. The core architecture depends on the concept of operating in multiple layers to form a multi-layered network, unlike usual blockchain architecture where peers all operate in the same network. The three layers that make up this framework are:

- *Smart home:* A smart home can be said to be a local network where there are a number of IoT devices and a central manager to manage these devices[35]. Devices communicate with each other through this manager and the manager itself maintains a local ledger and storage for these transactions.

- *Overlay Network:* The smart home managers (SHM) are connected to a peer-to-peer network, much like a traditional blockchain network with some differences. The SMH can act as a peer or any high-performance device connected to the SHM can take that role for the local network. These representatives of the smart home network are grouped into clusters and a coordinator is selected among them, henceforth known as the cluster head (CH). The cluster head can be selected randomly and also reassigned in case of problems or malicious activity. The role of the cluster head is to act as a certification authority as well as a peer of the blockchain network that stores and maintains the ledgers; one for each overlay node. Other cluster heads in the network can request data of specific overlay nodes according to the request of the nodes it is in charge of. The cluster head thus maintains a list of private keys for the requesters and requestees of data to ensure which node can access whose data. Each cluster head also maintains an individual private key for communication with other cluster heads. Block validation occurs using a PoW consensus protocol where all the transactions in a block are checked by all the cluster heads and the counter kept for accepted and rejected transactions is incremented appropriately, and the block is then broadcasted to all the nodes to update the ledger state. A trust rating is maintained by cluster heads to reduce computational overhead. Cluster heads with higher number of valid blocks have higher trust values and thus a block generated by these cluster heads has a fraction of the transactions verified, with higher trust values leading to lower number of transactions verified. Unlike the longest chain principle, forks are resolved by giving preference to blocks generated by CHs of higher trust.

- *Cloud Storage:* Once a block is verified, it is added to the ledger maintained for that specific overlay node in the cloud storage. The SHMs are all given a unique block identifier which is used by the cloud storage to store data as well as to access the local ledger of the SHM to extract data for storage. A homeowner can also create multiple ledgers if they wish to. The cloud storage guarantees the immutability of data which is not provided by the local ledger.

3.1.4 An efficient lightweight integrated blockchain (ELIB) model for iot security and privacy[6]

This[6] is similar to the [5] structure. The presented model contains two levels- smart home and overlay and the operation is divided into three parts: consensus algorithm, CC model, DTM scheme.

1. *Consensus Algorithm:* At first, overlay nodes get data from smart homes, which maintain a local ledger. In this regard, the authors' redefined the multi-sig transaction header for reducing overhead and increasing efficiency.

   The overlay nodes run a time-dependent consensus technique instead of PoW or PoS. In this method, the overlay nodes wait for arbitrary time and a single node is selected randomly to generate the block. To ensure that a malicious node is not trying to append a large block of false transactions, a short time count is provided which they called the consensus period.

2. *CC Method:* This is used for verifying the IoT devices using the key generation center (KGC). The working mechanism is -

   - KGC produces a partial private key using user A's id.
   - Each user has its secret value.
   - Partial private key and secret value create a private key for that user.
   - Each user generates its own public key using its secret value.

   The verification process is done in the following manner: When any IoT device posts a transaction with a signature signed using the private key, including a public key and its id, a miner checks whether that transaction signed with that private key is linked to a public key and whether that public key belongs to the device with that id.

3. *DTM:* DTM(distributed throughput manager) checks whether the consensus period should be increased or not by calculating a mathematical equation.

3.1.5 IoTLogBlock: Recording off-line transactions of low-power IoT devices using a blockchain[7]

To tailor a blockchain architecture for low-power, low memory-bandwidth and sparse network connectivity in IoT devices, [7] was developed with the concept of recording transactions offline until an edge device could be detected. It reduces the computation power required in the IoT devices, reduces the required storage, and also optimizes the architecture for LoRa WAN like LPWAN technologies[36].

IoTLogBlock works on three main principles to achieve this goal:

1. Smart contracts are programmed to execute alongside contract signing instead of contract signing followed by execution of smart contracts. The Asokan-Shoup-Waidner protocol[37] has been implemented for contract signing to ensure fairness and timeliness but with the catch of an offline Trusted Third Party which may be prone to security risks. Devices must also be registered by the MSP and assigned public/private key pairs before being able to partake in the contract signing process. Devices must maintain individual sequence numbers for every transaction number by incrementing a counter, and during a transaction, IDs, sequence numbers and transaction data must be exchanged and signed by both parties for a transaction to be considered valid.

2. Once the contract has been signed, both parties wait for a connection to an edge device to transfer their record of the transaction to the cloud, where it is processed by an implementation of the Hyperledger blockchain protocol. Transactions are then validated by another smart contract before updating the ledgers and world state.

3. Security is maintained by having each IoT device maintain individual sequence numbers for every transaction and by the contract signing protocol. Since both parties store a copy of the transaction and must upload them to the cloud, if one party tries to delete a transaction, it is detected by the cloud, the sequence number also prevents a device from skipping uploading a transaction. The MSP public/private key assignment further ensures the prohibition of entry to unregistered devices.

3.1.6 A scalable blockchain framework for secure transactions in IoT[8]

The authors[8] put forth a proposal similar to off-chains in [3] but here they replicate a Hyperledger Fabric network as a local network where peer nodes serve multiple IoT devices and each local network is connected to the main blockchain network through and anchor node, which can be of any blockchain architecture for true decentralization and scalability. For each local network, only the peer connected to the anchor can make audits and update the local ledger, the rest of the local peers only maintain a copy and serve IoT devices.

3.1.7 Non-linear adaptive closed-loop control system for improved efficiency in IoT-blockchain management[9]

The focus of the paper[9] is on the implementation of a networked control system (NCS) in IoT networks, with the authors have also introducing the concept of a queuing and hashing system for use in a blockchain network in order to tackle the communication limitation. The network can be broadly divided into three layers: the IoT layer, the blockchain layer and the big data layer. However, our concern is only the IoT layer and Blockchain layer for this paper.

- *IoT layer:* The IoT devices proposed have comparatively higher computing power, as seen with the example of Raspberry pis used for this paper. The authors have also added an extra miner processing unit. As a result, once the sensors connected to the raspberry pi generate data, the data is packaged and converted to a transaction with the help of a smart contract. This smart contract also helps to put the transactions in a block and the transactions as well as the block is verified before it is passed to the next layer.

- *Blockchain layer:* This layer consists of the usual blockchain nodes responsible for the consensus protocol and ledger. A queuing system exists between the IoT devices and the blockchain nodes, which manages traffic flow so as not to overwhelm the channel. Once the nodes receive a block, they pass a request back to their IoT devices for block validation as well as the mining for the PoW consensus. Once validated, the block is then added to the chain and the ledger is updated.

3.1.8 Design and implementation of an integrated IoT blockchain platform for sensing data integrity[10]

The authors in this paper[10] have implemented an architecture following the hyperledger framework. A few changes have been made, however. The IoT devices have been decoupled from the peer nodes and they exist purely as clients that submit their transaction requests to the main blockchain nodes. An additional server has also been added as an interfacing layer between users who want to access the data stored in the ledger and the main blockchain network.

3.1.9 Tikiri—towards a lightweight blockchain for IoT[11]

Like Hyperledger, the authors in [11] have tried to decouple the different functions from a single monolithic block as observed in popular blockchain systems like Bitcoin and Ethereum. They call these modular functions "Microservices"[38].

An understanding of these four services gives a complete picture of what the authors have tried to accomplish:

1. *Gateway Service:* Instead of having the client being a node that manages its transactions, the authors have used a gateway service API to submit transactions for execution. These transactions are then forwarded to the Kafka microservice in a JSON object.

2. *Lokka Service:* This service pulls transaction presented to the Kafka service and performs the following actions in order:

   (a) Transaction validation by checking signatures and timestamps

   (b) Smart contract execution

   (c) Block generation

   (d) Block broadcasting to other Lokka services for validation and consensus for updating the ledger

3. *Storage service:* The storage service is a group of nodes dedicated solely to storing the ledger state. Tikiri has the policy of setting a factor for storage to ensure that the same data is not replicated on all storages but in a fraction of them to ensure that the storage does not run out due to the possibility of large amounts of data being generated by IoT sensors.

4. *Kafka Service:* The Kafka service is also a consensus protocol and message broker that handles transaction transmissions from the gateway to Lokka services and also the communication between the Lokka services during block consensus.

3.1.10 A trust architecture for blockchain in IoT[12]

Following the research of the Authors[12] in developing the LSB model[39], they have proposed a solution much similar to what we have seen in [5] except the SMHs have been replaced with sensor nodes. The same two layered blockchain architecture is used with an Overlay network above the data sensing/generating nodes, but this time the novel concept of a trust based system has been integrated to ensure that the data collected is accurate and not manipulated.

Each sensor node must register itself into the network with its public key. Nodes close to each other with similar observations are grouped together and multiple nodes are assigned to a gateway node in the overlay layer which takes care of the computation for the blockchain layer. Every sensor and gateway node has a reputation value attached to it and every data collected has a trust value calculated for it by the gateway node its sensor is attached to. As there is no issue of the double spending problem, the consensus protocol is relatively simple where the transactions are verified using the public keys and recalculation of the trust value for each transaction. The trust based model works in three layers:

- Every data collected is assigned an individual trust value. This is calculated using a function of the reputation of the sensor node, the confidence of the data collected and the validity or trust in the data collected. The validity of the data collected by a node is calculated using the observations of similar nodes situated close to it which are supposed to give similar readings. The confidence refers to the uncertainty in the collected data.

- Every sensor node has a reputation, which increases with transactions with higher trust values and evidence of validation by other nodes in the cluster. Penalties are imposed for lower trust transactions. All the trust and reputation calculations are done by the attached gateway node.
- Every gateway node also has a reputation value. This is calculated and changed after a block is generated by the gateway node and the block has been validated by other gateway nodes by checking the signatures and recalculating the trust values. Honesty in block mining increases trust and dishonesty imposes penalties. With greater reputation values, the number of transactions that have to be validated in the block are reduced, thus reducing computational complexity.

3.2 Existing Consensus Protocols

3.2.1 Bitcoin: A Peer-to-Peer Electronic Cash System[1]

Bitcoin[1] first proposed the idea of proof-of-work(PoW). In a proof of work system, computers (called "miners") compete to solve a mathematical problem. The first miner to solve the problem gets to create a new block of transactions and add it to the blockchain, and in return they are rewarded with some units of the cryptocurrency. This process of solving the mathematical problem and creating new blocks is also called "mining."

The mathematical problem that the miners need to solve is designed to be difficult, so that it takes a lot of computing power to solve it. This ensures that new blocks are added to the blockchain at a regular rate and makes it difficult for any single miner to gain control of the blockchain[40].

Many cryptocurrencies, including Bitcoin, use proof of work to secure the blockchain and validate transactions. It has proven to be a reliable and effective approach, but it does have certain disadvantages, such as the significant energy consumption required for mining.

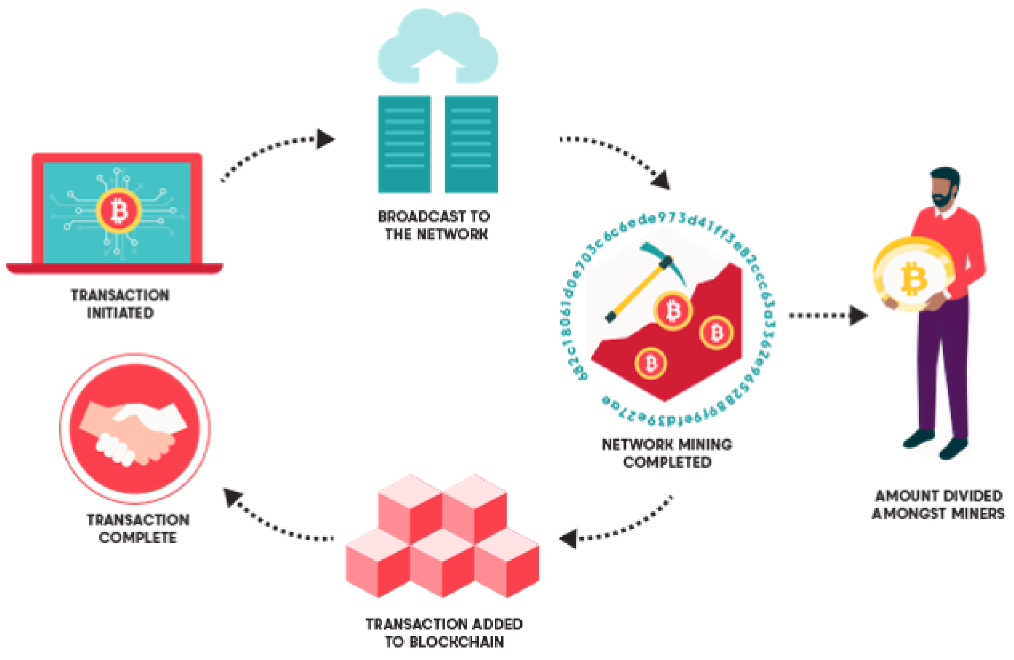


Figure 4: Proof-of-Work

3.2.2 Proof-of-Stake Sidechains[13]

A proof of stake (PoS) method selects the creator of a new block in a deterministic manner based on their stake (how many coins they own) in the cryptocurrency. A new block's inventor is chosen in proof of work based on their ability to solve a mathematical puzzle.

In a PoS system, a miner must demonstrate that they possess a specific quantity of coins and ”lock up” or ”stake” those coins as security in order to create a new block. The action is referred to as ”staking.” A miner's chances of being selected to build the next block and receive the reward increase with the number of coins they bet.

Proof of stake has several advantages over proof of work, including being more energy-efficient because miners are not required to carry out computationally demanding activities. Since the capacity to create new blocks is not dependent on the speed of mathematical computation, PoS systems also tend to be more decentralized.

Proof of stake systems, however, are susceptible to some assaults, including "nothing at stake" attacks, in which miners wager on many chains simultaneously, and "long-range" attacks, in which an attacker can fabricate a blockchain with a longer history than the real one. Several approaches and procedures can be used to reduce the impact of these attacks.

3.2.3 On Security Analysis of Proof-of-Elapsed-Time (PoET)[14]

A consensus procedure called Proof of Elapsed Time (PoET) is employed in distributed systems to come to an understanding on the sequence of events. It is comparable to proof of work (PoW) in that it requires miners to accomplish a specific amount of computing effort, but it differs in that the next miner to create a new block is chosen using a random wait time rather than a challenging mathematical problem.

In a PoET system, a dependable party known as a "validator" must grant miners a wait time request. Each miner receives a wait time from the validator, and the next block is created by the first miner to complete their given wait time. We refer to this process as "winning the lottery." Every miner, regardless of computer capacity, has an equal chance of winning the lottery because the wait times are chosen at random.

One of the main advantages of PoET is that it is highly energy-efficient since it does not require miners to perform computationally intensive tasks. It is also highly scalable, since the amount of work required to win the lottery is constant, regardless of the number of miners. PoET is used by the Hyperledger Sawtooth platform, among others.

3.2.4 Delegated Proof of Stake[15]

Delegated proof of stake (DPoS) is a type of consensus algorithm used in some blockchain systems to achieve distributed consensus. In a DPoS system, there are two types of participants: delegates and stakeholders. Stakeholders are individuals or organizations that hold a stake (a certain number of coins) in the cryptocurrency, and they have the right to vote for delegates. Delegates are chosen by the stakeholders to represent them and create new blocks on the blockchain.

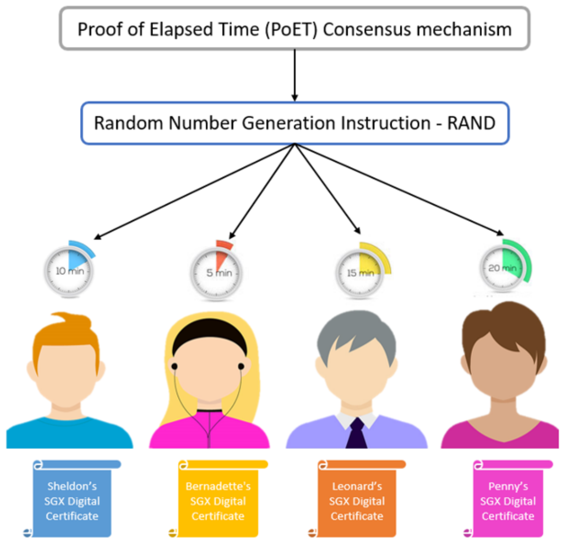

Proof of Elapsed Time (PoET) Consensus mechanism
Random Number Generation Instruction - RAND
10 min
5 min
15 min
20 min
Sheldon's SGX Digital Certificate
Bernadette's SGX Digital Certificate
Leonard's SGX Digital Certificate
Penny's SGX Digital Certificate

Figure 5: Proof-of-Elapsed-Time

Unlike in proof of work (PoW) or proof of stake (PoS) systems, where new blocks are created by miners who perform certain computations or hold a certain number of coins, in a DPoS system, new blocks are created by the delegates who are chosen by the stakeholders through a voting process. The delegates are responsible for validating transactions and adding them to the blockchain.

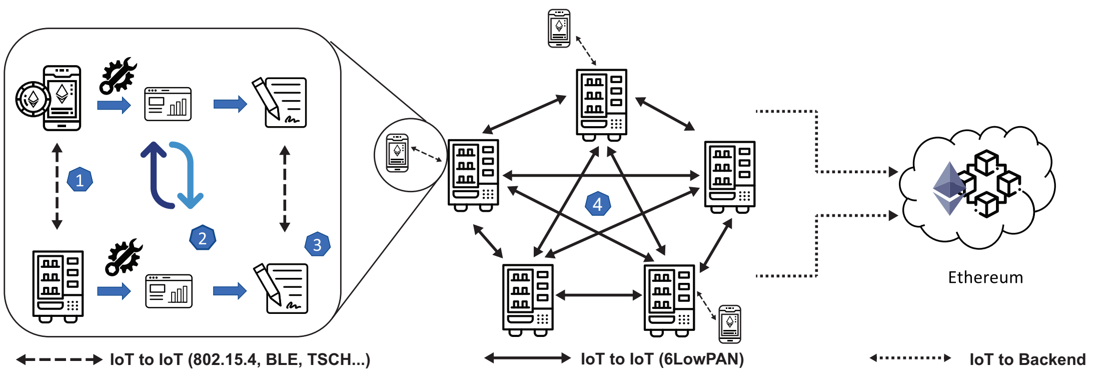
DPoS systems are designed to be faster and more efficient than PoW or PoS systems, since the number of delegates is typically small and fixed, and the delegates are chosen based on their reputation and ability to represent the interests of the stakeholders. However, DPoS systems can be vulnerable to centralization, since the power to create new blocks is concentrated in the hands of a small group of delegates. DPoS is used by several cryptocurrencies, including EOS and Steem.

3.2.5 DT-DPoS: A Delegated Proof of Stake Consensus Algorithm with Dynamic Trust[16]

DT-DPoS[16] is the upgraded version of the DPoS. The system requirement remains exactly the same as DPoS. The only change in this consensus is the introduction of trust ratings for the witness nodes. The trust rating can be used to select trustworthy nodes for the next round.

4 PPoS: Overview

This section uses a hypothetical scenario to express the motivation behind our consensus mechanism. We show our reasoning for the application of side-chains in TinyEVM specifically, explore and reason against other possible consensus protocols based on efficiency and security.



4.1 Application Scenario: Vending Machines

Vending Machines have always been widely adopted throughout the world due to the efficiency and autonomy of operating these machines. While even a few years back vending machines had been used primarily to stock and sell low-priced goods such as cheap drinks and snacks, with the advent of more secure and intelligent constructs, these machines are now used to sell a variety of goods. Recent advancements have allowed the use of mobile banking and even credit and debit cards to make purchases using these machines; all of which is done wirelessly with the machines being powered by battery systems.

We expect that with the increasing adoption of cryptocurrency use in many countries, vendors would be more inclined to add another option viable to make transactions: public cryptocurrencies like Bitcoin and Ethereum.

Customers willing to make purchases would be required to have a device capable of communicating and storing data using standard interfaces and systems along with being capable of running TinyEVM. Customers need not require a connection to the internet as initiating a transaction can be done through the machine. Upon initiation through low power near field communication channels, the machine would forward a copy of the smart contract to the customer's device and upon reaching an agreement, funds would need to be locked as a security. Subsequent transactions can be carried out between the customer and any other machine connected to the network through offline channels and only the exit would require another connection to the main blockchain.

4.2 System Requirements

To achieve the mentioned scenario, our system would need to have the following requirements:

Multi-party transactions: TinyEVM, while it solves the problem of implementing public blockchains in low-powered IoT devices, is designed primarily for communication between two parties. We thus need to extend it to facilitate transactions between multiple parties concurrently.

Energy Efficient: Due to constraints imposed by battery systems, our consensus mechanism need to be efficient to complement the benefits provided by TinyEVM. Computing power, wireless transmissions and network traffic are the major factors affecting efficiency in our system.

Secure: As the user can make multiple transactions throughout the network of devices, we need to provide a system to securely keep track and ensure fairness throughout the process.

4.3 Security Challenges

We assume that both the vendor and the customer would threaten the model through repudiation of transactions. The customer faces the uncertainty of not receiving the locked funds or being charged unfairly for items not received. The vendor faces the threat of falsified funds when making offline transactions across multiple nodes.

5 PPoS: System Design

In this section, we go over the key phases that make our consensus mechanism viable. Our mechanism relies heavily on some of the core features proposed by TinyEVM, such as the off-chain payment channels, on-chain commits and the use of a nonce. However, as all of these concepts are not limited to TinyEVM, our consensus mechanism can also be applied to more general applications. The core four phases that make up our mechanism are: a) Network Setup, b) Transaction Execution, c) Validation and d) Online Commits. When describing these phases, we make use of a few terms: Nodes, Validators, Supervisors and Users.

Nodes: A node is any IoT device that is capable of running TinyEVM and has wireless connectivity to participate in the blockchain network. Al of the participating IoT devices in our network can be considered to be a node, with the exception of Users. For example, in our proposed scenario, all of the vending machines can be considered to be nodes.

Supervisors: Supervisors are a special subset of Nodes selected to act as the primary means of communication between the main Ethereum Blockchain and the side-chain running the consensus mechanism. They have a number of responsibilities, which consist of recording Transactions, selecting Validators, cross-checking Block validity and finally uploading Transactions to the main chain.

Validators: Validators are another subset of Nodes existing apart from Supervisors. They are responsible for checking the validity of transactions and creating blocks to be added to the main chain. Validators are selected through a system of votes and some additions parameters from ordinary Nodes.

Users: A User is any entity that carry out one or more transactions with the Nodes in our blockchain network. They can be IoT or mobile devices operated by humans or machines that can interact with the Nodes and have sufficient monetary assets to take part in transactions. They do not directly participate in the blockchain network at any point.

5.1 Network Setup

In most other consensus protocols, there is virtually no need for any network setup phase as for both guided and unguided media, consensus protocols generally rely on broadcasting transactions to all participating nodes in order to ensure security and integrity. PPoS, however, aims to multi-cast transactions to relevant parties instead of broadcasting, which solves a host of inefficiencies such as unwanted collisions, duplicate packets and more in wireless media. As a result, we need an extra step to set up our network before nodes can participate in transactions. There are three sub-phases that encompass this network setup phase:

5.1.1 Side-chain Initiation

Once the smart contract for the side-chain has been published, Nodes in the network download and forward a copy to neighboring nodes until all of the nodes have received a copy and become ready to execute the smart contract. All the nodes in the network exist as a part of a singular entity so any transactions or contracts carried out between a Node and a User is fundamentally taking place between the whole side-chain and the User.

5.1.2 Multi-cast Group Setup

For each and every facet of communication within the side-chain, our consensus will rely on multi-casting packets instead of broadcasting. As a consequence, every Node needs to contain its own multi-cast routing table for Supervisors and Supervisors additionally also need to have a multi-cast routing table for Validators. For every round of Supervisor Selection and Validator Election, as well as the addition of any new Node into the network, these multi-cast routing tables would also need to be updated accordingly. Considering that Ethereum natively has no support for multi-cast routing, both multi-cast and unicast routing protocols have to be implemented using smart contracts or added to the TinyEVM code, and MOSPF would be the simplest to implement while ensuring security and integrity.

5.1.3 Supervisor Selection

To ensure the integrity of the network and to prevent power being concentrated among a select few Nodes in the blockchain network, Supervisors will need to be re-selected every few rounds after the intial selection. This selection is to be done using a function that utilizes a combination of three factors: a random hash function, a trust value and a counter to prevent frequent re-elections of a single Node. Every Supervisor Selection round is followed by a Multi-cast Group Setup and only after the Selection can waiting new Nodes join the network to participate in the consensus mechanism.

5.1.4 Validator Election

Validator Election occurs in every round of block formation to ensure that all participating Nodes have the opportunity not only gain block rewards but to also prove and improve their trust values. Validators are selected by Supervisors based on the amount of coins staked by other Nodes on their behalf. No Node can stake for itself and must stake for any other Node. Block rewards are distributed among the Validators and the Nodes staking for the selected Validators.

5.2 Transaction Execution

Once the Supervisor Selection has taken place, Users are free to carry out transactions with any of the Nodes in the network. Before making any transaction, however, the User needs to deposit a certain amount of funds and lock it into to side-chain. Once the funds have been deposited, the User receives a signed certificate from the Node it is communicating with on behalf of the side-chain as proof of the remaining funds. Subsequent transactions with any of the Nodes in the network require this proof to be presented and exchanged for a new one to demarcate the most recent state of funds. Once the user is ready to close their transactions with the side-chain, the last proof needs to be signed by the User and presented to any Node to mark the exit and hence unlocking and transfer of funds. All forms of communication between a Node and a User occur through off-chain channels with only the initial locking of funds requiring the corresponding Node to communicate with the main chain.

Transactions are not broadcasted to the network immediately as is the case with most other consensus protocols. Instead, they are stored and accumulated by the participating Nodes until a numerical imposed limit is reached or the next validation phase starts. Once the limit is reached or the validation phase starts, all the accumulated transactions and last proof of states are multi-casted to the Supervisors.

5.3 Validation

The Validation phase requires the participation of all three types of Nodes in the side-chain. Supervisors first receive and store the transactions that took place after the last validation phase. Once all Nodes are done transmitting their transactions along with their votes for Validators, the Supervisors select the Validators and forward the transactions to them for block formation. Validators take these transactions and verify them, adding only the final state of the completed chains of transactions to the block. The incomplete chains have their last state verified and the history of transactions discarded to save space. The newly formed block and list of incomplete transactions are then transmitted back to the Supervisors for cross-validation.

5.4 Online Commits

Following the cross-validation of the received data, the Supervisors upload the final states of the complete transactions in order to unlock and receive the deserved funds. The proof has the signature of both the User and the last Node they interacted with, and a local log of transactions is kept with the participating Nodes to provide a window for either party to challenge the proof until the next round of validation. The remaining fund is returned to the User if there is no contest.

6 Consensus Mechanism

In the previous section, we gave a top-level view of our consensus protocol. Here, however, we will go into detail about the inner workings that make our consensus feasible and the improvements and/or adjustments done to tailor it for our envisioned scenario.

6.1 Transmission media and Protocols

In our proposed system, all communications within the network occur using the 6LoWPAN protocol stack. While alternatives such as LPWAN and BLE may exist, they have their own disadvantages in this application. LPWAN protocols provide a much better battery life but conversely, their data rate is abysmally low compared to that of BLE and 6LoWPAN. BLE on the other hand has both better battery life and similar data transmission rates but requires a star or mesh topology which greatly compromises the distributed aspect of a blockchain network. Thus 6LoWPAN provides a better trade-off as it supports a mesh topology while providing high data rates at reasonable battery efficiencies. Communication between a User and a Node, however, can be easily taken care of by BLE or NFC. Finally, the on-chain commits require a strong, stable long-range connection such as 4G or 5G mobile networks, which are extremely battery intensive, thus the choice of having Supervisors being solely responsible for them.

6.2 Setup, Selection, and Election

When setting up the side-chain network initially, all the Nodes download a copy of the smart contract template and then exchange their information (certification, identification information etc.); the trust and coin values are set to zero. Every Node thus has a list of all the other Nodes in the network with their corresponding public keys, trust values, repetition values and identification. The Nodes then proceed to the Selection phase. However, whenever a new Node wants to join the network, it has to wait until the end of the next selection phase. The new Node(s) is provided information such as trust values and identification information of all the nodes in the network by all the Supervisors only. This serves to not only reduce traffic but also ensure that accurate information is conveyed as Supervisors are generally trustworthy parties.

During the Selection phase all the participating Nodes first use a pseudo-random function generator to select a random number. The random number is the hashed using a generic hashing function, along with the hash of the private key of the Node. The first 64 bits of both these hash values are added and hashed again to generate a new random hash, which is used to calculate the hamming distance from the hash of the node's public key.

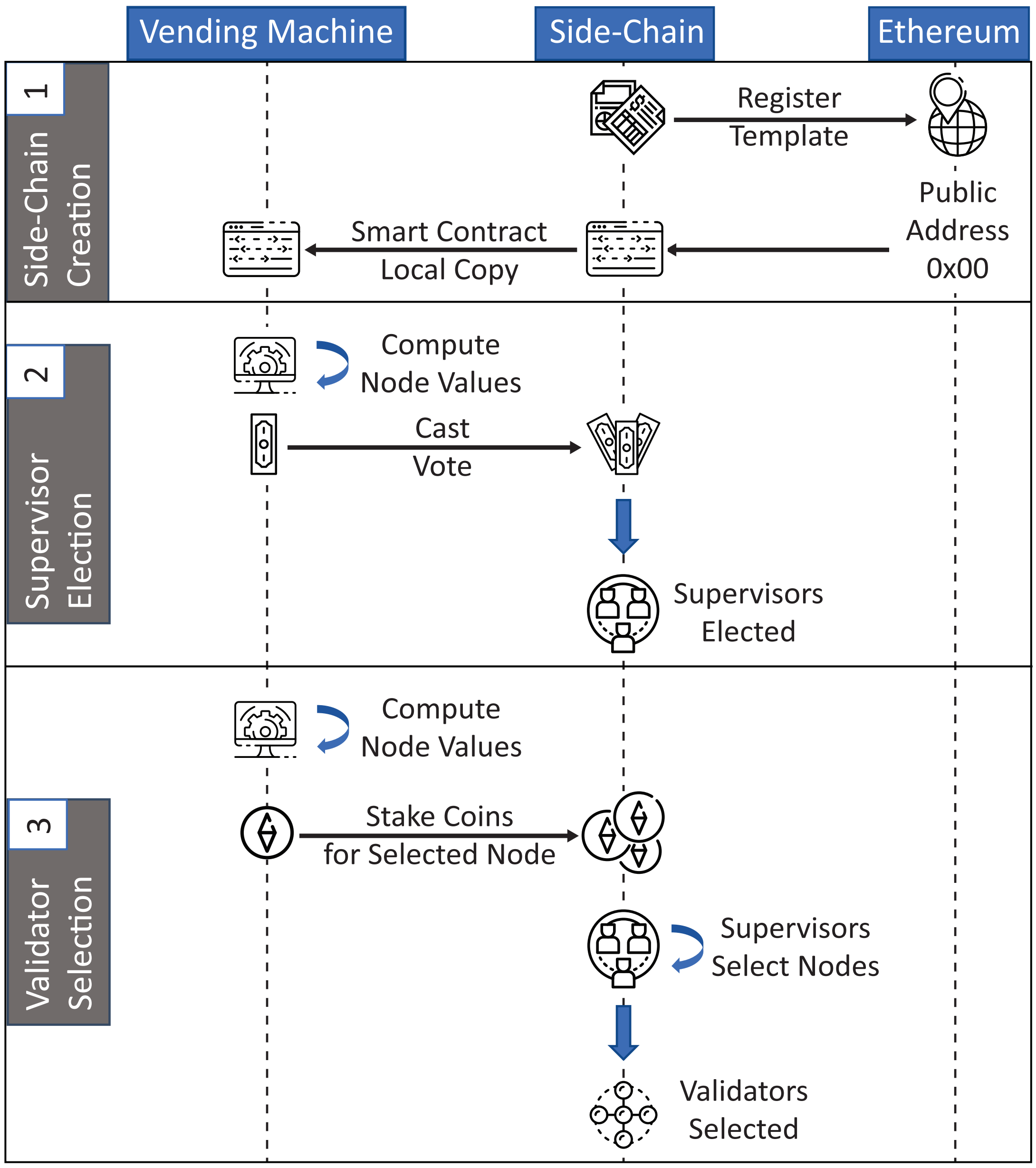


Figure 6: Sequence Diagram of Setup, Selection & Election Process

This hamming distance, along with the trust and repetition values are then normalized to 1 and used to cast a vote. We should be aware that there are separate repetition values for Supervisors and Validators and for every successful selection, the values are subjected to an arithmetic right shift operation to reduce the chances of being re-elected repeatedly. Three variables, α, β and γ are used as a scaling factor for each of the values used to calculate the vote in order to allow for flexibility in prioritizing between the choosing factors. The votes are multi-casted to the previous Supervisors so that they can inform the new batch. The new Supervisors then broadcast their identities to both inform the network as well as to create new multi-cast groups. Once the Supervisors

Algorithm 1 Supervisor Selection

```
Require: $nodeList[\,], key, maxTrust, \alpha, \beta, \gamma$
1: $r = rand()$
2: $r = hash(r) + hash(key)$ //First 64 bits only
3: $selNode = $ NULL
4: $maxVal = 0$
5: for all $node$ in $nodelist[\,]$ do
6:     $val = \alpha * hammingDist(r, hash(node.key))$
7:     $val$ += $\beta * (node.trust/maxTrust)$
8:     $val$ -= $\gamma * (node.repeat/2^{32})$
9:     if $val > maxVal$ then
10:        $maxVal = val$
11:        $selNode = node$
12:    end if
13: end for
14: $Vote(selNode)$
```

have been selected, they each choose a random number from 1 to 100 and decide on an order of signing the completed block that is to be sent for the on-chain commit. If there are conflicts when selecting the order, the conflicting Nodes carry out multiple rounds until the conflict is resolved. The previous Supervisors then transmit the record of all incomplete transactions to the new ones.

The Election phase takes place using a similar process but with two minor changes: instead of computing the hash, a random number is selected between 0 and 100 instead to reduce the cost of computation and instead of votes, coins are staked by the voting Nodes. Note that the Nodes cannot stake coins for themselves. Supervisors then order the Nodes by the amount of coins staked and select the top few as Validators.

The trust value is incremented for every selected Validator after each validation round. If the Validator is instead caught to be misbehaving, the Node is penalized by having the trust value subject to a left shift. This means that nodes with higher trust values are penalized more, as they are more likely to be selected.

Algorithm 2 Validator Election

```
Require: $nodeList[\,], maxTrust, \alpha, \beta, \gamma$
1: $selNode$ = NULL
2: $maxVal = 0$
3: $finalR = 0$
4: for all $node$ in $nodelist[\,]$ do
5:     $r = rand(0, 100)/100$
6:     $val = r + \beta * (node.trust/maxTrust)$
7:     $val$ -= $\gamma * (node.repeat/2^{32})$
8:     if $val > maxVal$ then
9:         $maxVal = val$
10:        $selNode = node$
11:        $finalR = r$
12:    end if
13: end for
14: $Stake(selNode, finalR)$
```

6.3 Trade Initiation, Execution, and Completion

Any User willing to take part in a transaction must first have an account or public key registered in the Ethereum blockchain. To initiate, a User can simply approach any Node in the network and download or receive a copy of the appropriate smart contract template. The User must then agree upon a certain amount of funds to be locked into the side-chain before making any transactions. The whole process can be carried out without the need of any internet connection on behalf of the User as the Node can act as an intermediary. Once the funds have been locked, the User receives a signed proof of their unutilized funds which is required to carry out subsequent transactions. The User then returns a copy of the proof with their own signature to the Node.

Every time the User initiates a new transaction, the corresponding Node first checks the balance using the proof, checks with the Node that signed the proof and then proceeds, providing the User with a new proof. The previous proof is then signed by the transacting Node followed by the User. This ensures that that User cannot fake funds and the Nodes cannot submit duplicate proofs as a challenge.

To complete the series of transactions, the User must exchange a final signed agreement with any of the Nodes in the network. The User or network can then simply submit the summary of transactions with the proofs to the main chain to initiate a challenge period that is twice the length of the interval between validation rounds. This is to ensure that both parties have adequate time to check and respond to challenges.

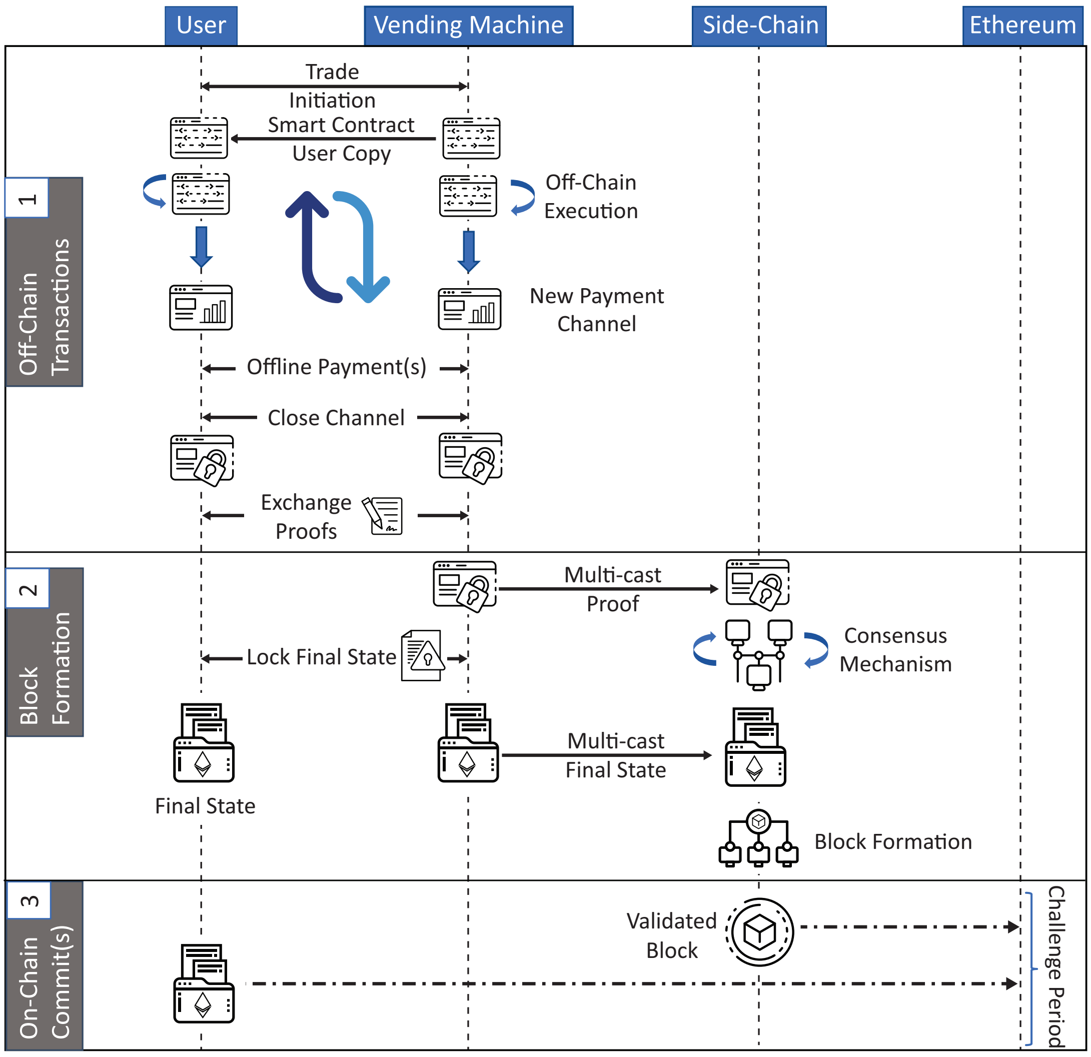


Figure 7: Sequence Diagram of Trade Initiation, Execution & Completion Process

6.4 Block Formation and Submission

Once the election round has ended and the Validators receive the list of transactions through multi-casting from the Supervisors, they start verifying the transactions by checking the digital signatures of the participants and the sequence number attached to each transaction. Transactions are ordered first by completion status and then by the amount of remaining funds, ensuring that all the Validators create the same block. Every block is signed by the respective Validator who created it as an identifier and then forwarded to the Supervisors again through multi-casting.

Supervisors receive these blocks and cross-check the copies for fraud. Once done, they remove the incomplete transactions and start the process of signing the block hash in the predetermined order. After all the Supervisors have signed the block hash, the copy is sent to all of them to verify the signed hash. Upon determining the authenticity, all the Supervisors transmit the block to the main chain through any long-range high data rate wireless technology such as 4G or 5G. Any Validator of the main chain, before adding the transaction of the main chain, verify the authenticity of the block through the signed block hash, as a part of the procedure stated in the smart contract.

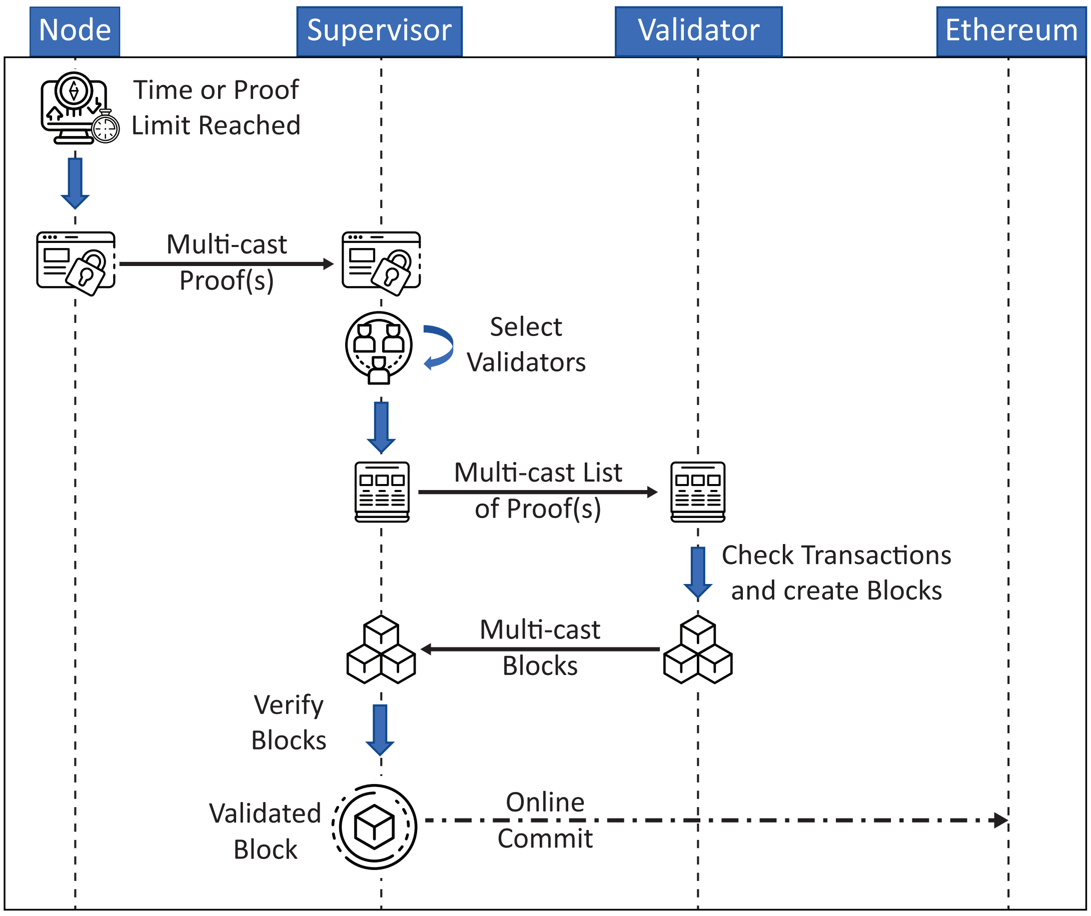


Figure 8: Sequence Diagram of Block Formation & Submission Process

7 Security Analysis

Unlike TinyEVM, our consensus protocol has to actively implement measures to prevent double-spending and other fraudulent behavior, similar to multi-party consensus protocols. To allow continuous micro-transactions without having to actively communicate to the whole network every time a transaction is carried out, our system makes use of a few key concepts:

1. The use of a double sequence counter, one for the user and one for the node,
2. Periodically changing the Validators and Supervisors to prevent power concentration,
3. Use of a Trust rating to track misbehaving parties and
4. Punishments and rewards to discourage misbehavior.

The threats to our system can be categorized into two types: Threats by Users and threats by vendors.

User Threats: Users may try to falsify funds in order to make transactions beyond the capacity of funds deposited. They may also try and deny some of the transactions during the contention period to regain the locked funds even after consuming the goods or services of the vendor.

Vendor Threats: Vendors may try to duplicate transactions or compromised Nodes may try and delete records of transactions between validation rounds.

7.1 Verification of Transactions

Every exchange carried out between a User and Node has two sequence numbers attached and the date of the transaction along with a proof stating the remaining funds. Copies are kept by both parties and this prevents and double-spending or deletion of transactions as the honest party can easily submit the proof of transactions during the transaction or challenge period to seize the locked funds. As a result, both parties are motivated to behave honestly both to not be penalized and to be rewarded in case the other party misbehaves.

7.2 Validation of Blocks

Validators are required to check the authenticity and proof of funds for each transaction before cross-checking with the final proof to ensure integrity. As both the proofs and transactions are signed by both participants, they can be easily verified, and produced blocks are signed by the Validators. Due to the presence of multiple Validators and the resulting blocks being cross-checked by Supervisors, compromised blocks and Validators can be easily identified and penalized with the loss of the staked coins and a reduction in Trust value. Nodes that had cast votes for the misbehaving Validator are thus also penalized through the loss of funds and consequently motivated to make better choices in the future. Compromised Supervisors are detected and penalized through Trust values by Validators when the same data is not received from all of the Supervisors.

7.3 Integrity of Commits

Co-signing a select validated block by all of the Supervisors and simultaneous commit by all of them ensures integrity as any misbehaving Supervisor will be detected the moment of refusal of signing or uploading a modified block as a unique order of signage is established every Supervisor Selection round. In case of a compromised Supervisor, their trust value is reduced and a new selection round is immediately carried out.

8 Performance Evaluation

Comparing the consensus protocols, it can be observed that PPoS can efficiently decrease the number of transmissions during the consensus period. To validate our claim, a set of simulations are done in Bitcoin Simulator[41] by simulating the existing consensus protocols as well as PPoS.

A set of assumptions has been made beforehand the simulation of these consensus protocols. Those are-

- Each interior node is connected to 4 other nodes.
- Packet sizes are not considered.
- No packet losses or queuing delays are considered.
- DPoS and PPoS have 3 Supervisor and 5 validator nodes.
- Block Size = 100.

After simulation, the following result is produced-

Table 1: Number of Transmissions for Different Consensus Mechanisms

Number of Nodes	**PPoS**	**DPoS**	**PoS**	**PoW**
20	312	3619	2052	1356
30	453	4729	2945	2184
40	586	5958	4041	2917
50	719	7125	4913	3561
100	1426	13498	9574	7432
200	2148	27143	19580	14926
300	2564	32649	29752	20167
400	2966	40706	37159	28139
500	3215	49618	48925	36148

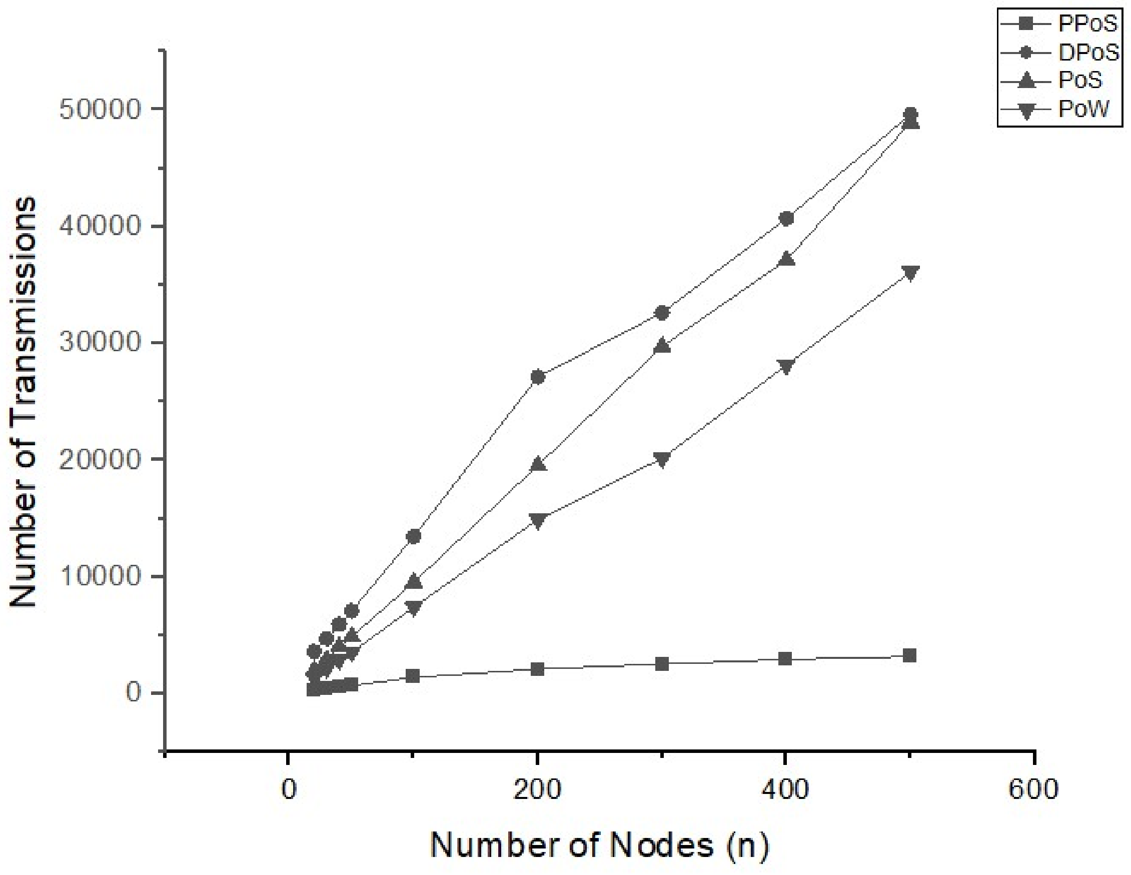


Figure 9: Comparison of different consensus algorithms

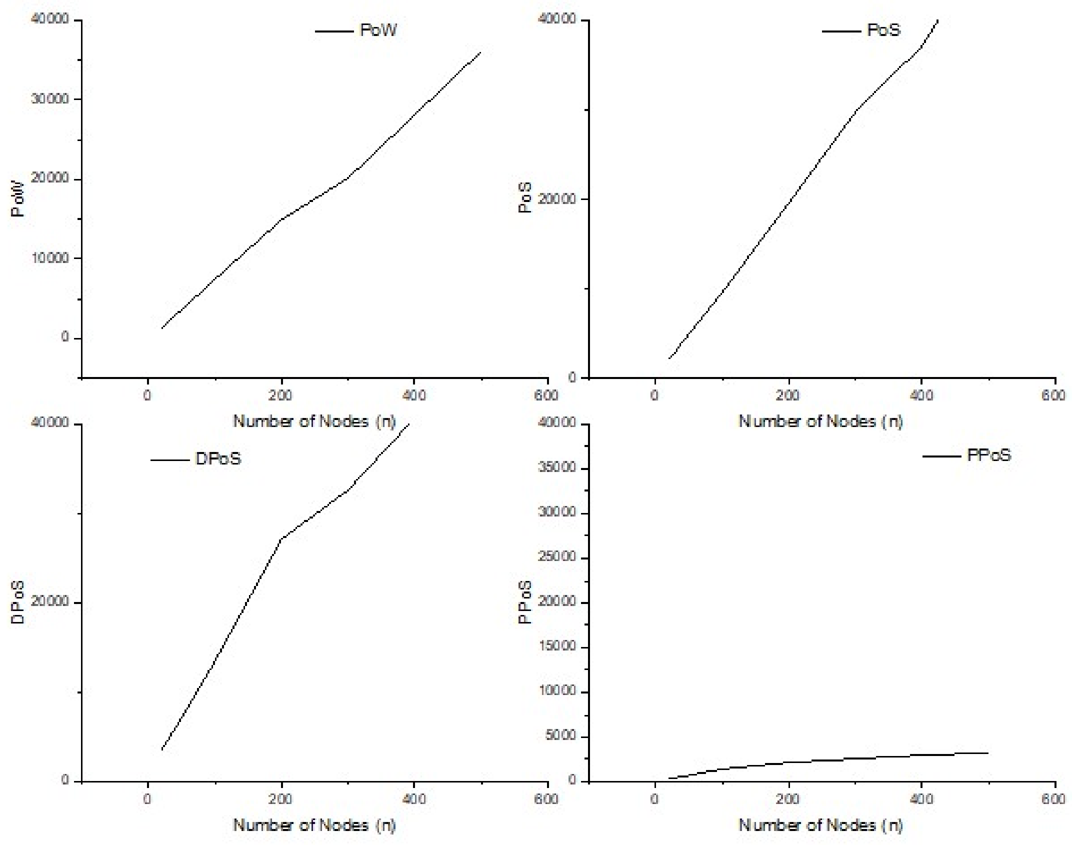


Figure 10: Comparison of different consensus algorithms (2)

The table provides insights into the relationship between the number of nodes and the number of transmissions for different consensus mechanisms, highlighting the variations and differences in their performance. From the generated data, it is evident that-

- As the number of nodes increases, the number of transmissions generally increases for all consensus mechanisms.
- Across all node counts, the PoW mechanism consistently has the highest number of transmissions, followed by PoS, DPoS, and PPoS.
- The PPoS mechanism has the lowest number of transmissions among the four mechanisms.
- The number of transmissions for each mechanism varies non-linearly as the number of nodes increases.

With this efficiency, PPoS surpasses existing consensus mechanisms for IoT devices in terms of optimizing the network flow. Thus we believe our system can be deployed for permissionless public blockchains for IoT devices.

9 Conclusion & Future Works

Unlike the existing consensus mechanisms for permissionless public blockchains for IoT devices, our proposed consensus protocol PPoS can efficiently generate a way less number of transmissions which inevitably reduce the network traffic during the consensus period. Our proposed protocol works similarly to DT-DPoS, providing more flexibility in choosing witness nodes and preventing malicious nodes from being selected as witness nodes.

However, our proposed protocol can be tested outside of the scope of our initial assumptions. Thus our future interests are enumerated as below-

- Implementation considering payload and different validator-supervisor ratios.
- Generation of optimal α, β, & γ values using machine learning techniques for specific network configurations.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[2] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[3] C. Profentzas, M. Almgren, and O. Landsiedel, "Tinyevm: Off-chain smart contracts on low-power iot devices," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS).* IEEE, 2020, pp. 507–518.

[4] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, "Pobt: A lightweight consensus algorithm for scalable iot business blockchain," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2343–2355, 2019.

[5] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for iot," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI).* IEEE, 2017, pp. 173–178.

[6] S. N. Mohanty, K. Ramya, S. S. Rani, D. Gupta, K. Shankar, S. Lakshmanaprabu, and A. Khanna, "An efficient lightweight integrated blockchain (elib) model for iot security and privacy," *Future Generation Computer Systems*, vol. 102, pp. 1027–1037, 2020.

[7] C. Profentzas, M. Almgren, and O. Landsiedel, "Iotlogblock: Recording off-line transactions of low-power iot devices using a blockchain," in *2019 IEEE 44th Conference on Local Computer Networks (LCN).* IEEE, 2019, pp. 414–421.

[8] S. Biswas, K. Sharif, F. Li, B. Nour, and Y. Wang, "A scalable blockchain framework for secure transactions in iot," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4650–4659, 2018.

[9] R. Casado-Vara, P. Chamoso, F. De la Prieta, J. Prieto, and J. M. Corchado, "Non-linear adaptive closed-loop control system for improved efficiency in iot-blockchain management," *Information Fusion*, vol. 49, pp. 227–239, 2019.

[10] L. Hang and D.-H. Kim, "Design and implementation of an integrated iot blockchain platform for sensing data integrity," *Sensors*, vol. 19, no. 10, p. 2228, 2019.

[11] E. Bandara, D. Tosh, P. Foytik, S. Shetty, N. Ranasinghe, and K. De Zoysa, "Tikiri—towards a lightweight blockchain for iot," *Future Generation Computer Systems*, vol. 119, pp. 154–165, 2021.

[12] V. Dedeoglu, R. Jurdak, G. D. Putra, A. Dorri, and S. S. Kanhere, "A trust architecture for blockchain in iot," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2019, pp. 190–199.

[13] P. Gaži, A. Kiayias, and D. Zindros, "Proof-of-stake sidechains," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 139–156.

[14] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)," in *International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Springer, 2017, pp. 282–297.

[15] Dpos consensus algorithm - the missing white paper. [Online]. Available: https://steemit.com/dpos/@dantheman/dpos-consensus-algorithm-this-missing-white-paper/

[16] Y. Sun, B. Yan, Y. Yao, and J. Yu, "Dt-dpos: a delegated proof of stake consensus algorithm with dynamic trust," *Procedia Computer Science*, vol. 187, pp. 371–376, 2021.

[17] Internet of things. [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_things

[18] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1–15.

[19] Sidechains. [Online]. Available: https://ethereum.org/en/developers/docs/scaling/sidechains/

[20] Off-chain transaction. [Online]. Available: https://www.ledger.com/academy/glossary/offchain

[21] T. B. Malla, A. Bhattarai, A. Parajuli, A. Shrestha, B. B. Chhetri, and K. Chapagain, “Status, challenges and future directions of blockchain technology in power system: A state of art review,” *Energies*, vol. 15, no. 22, p. 8571, 2022.

[22] B. Shrimali and H. B. Patel, “Blockchain state-of-the-art: architecture, use cases, consensus, challenges and opportunities,” *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 9, pp. 6793–6807, 2022.

[23] K. Azbeg, O. Ouchetto, S. Andaloussi, and L. Fetjah, “A taxonomic review of the use of iot and blockchain in healthcare applications,” *Irbm*, vol. 43, no. 5, pp. 511–519, 2022.

[24] J. R. Bhat, S. A. AlQahtani, and M. Nekovee, “Fintech enablers, use cases, and role of future internet of things,” *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 1, pp. 87–101, 2023.

[25] M. A. I. Mozumder, M. M. Sheeraz, A. Athar, S. Aich, and H.-C. Kim, “Overview: Technology roadmap of the future trend of metaverse based on iot, blockchain, ai technique, and medical domain metaverse activity,” in *2022 24th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2022, pp. 256–261.

[26] R. Kumar and R. Sharma, “Leveraging blockchain for ensuring trust in iot: A survey,” *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 10, pp. 8599–8622, 2022.

[27] A. Alkhateeb, C. Catal, G. Kar, and A. Mishra, “Hybrid blockchain platforms for the internet of things (iot): A systematic literature review,” *Sensors*, vol. 22, no. 4, p. 1304, 2022.

[28] R. Chaganti, B. Bhushan, and V. Ravi, “A survey on blockchain solutions in ddos attacks mitigation: Techniques, open challenges and future directions,” *Computer Communications*, 2022.

[29] R. Verma, N. Dhanda, and V. Nagar, "Security concerns in iot systems and its blockchain solutions," in *Cyber Intelligence and Information Retrieval: Proceedings of CIIR 2021.* Springer, 2022, pp. 485–495.

[30] Y. Xiao, S. Shi, W. Lou, C. Wang, X. Li, N. Zhang, Y. T. Hou, and J. H. Reed, "Decentralized spectrum access system: Vision, challenges, and a blockchain solution," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 220–228, 2022.

[31] Z. Shah, I. Ullah, H. Li, A. Levula, and K. Khurshid, "Blockchain based solutions to mitigate distributed denial of service (ddos) attacks in the internet of things (iot): A survey," *Sensors*, vol. 22, no. 3, p. 1094, 2022.

[32] Ethereum virtual machine (evm). [Online]. Available: https://ethereum.org/en/developers/docs/evm/

[33] R. Antwi, J. D. Gadze, E. T. Tchao, A. Sikora, H. Nunoo-Mensah, A. S. Agbemenu, K. O.-B. Obour Agyekum, J. O. Agyemang, D. Welte, and E. Keelson, "A survey on network optimization techniques for blockchain systems," *Algorithms*, vol. 15, no. 6, p. 193, 2022.

[34] S. Tanwar, N. Gupta, C. Iwendi, K. Kumar, and M. Alenezi, "Next generation iot and blockchain integration," *Journal of Sensors*, vol. 2022, 2022.

[35] Z. Rahman, X. Yi, S. T. Mehedi, R. Islam, and A. Kelarev, "Blockchain applicability for the internet of things: Performance and scalability challenges and solutions," *Electronics*, vol. 11, no. 9, p. 1416, 2022.

[36] M. Jouhari, N. Saeed, M.-S. Alouini, and E. M. Amhoud, "A survey on scalable lorawan for massive iot: Recent advances, potentials, and challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[37] N. Asokan, V. Shoup, and M. Waidner, "Asynchronous protocols for optimistic fair exchange," in *Proceedings. 1998 IEEE Symposium on Security and Privacy (Cat. No. 98CB36186).* IEEE, 1998, pp. 86–99.

[38] K. Bantia, N. Govardhan, M. Anala *et al.*, "Inter-service communication among microservices using kafka connect," in *2022 IEEE 13th International Conference on Software Engineering and Service Science (ICSESS).* IEEE, 2022, pp. 43–47.

[39] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Lsb: A lightweight scalable blockchain for iot security and anonymity," *Journal of Parallel and Distributed Computing*, vol. 134, pp. 180–197, 2019.

[40] S. G. Motlagh, J. Mišić, and V. B. Mišić, "The impact of selfish mining on bitcoin network performance," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 724–735, 2021.

[41] A. Gervais, G. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 23nd ACM SIGSAC Conference on Computer and Communication Security (CCS)*. ACM, 2016.