



HPoC: A Lightweight Blockchain Consensus Design for the IoT

Zixiang Nie * , Maosheng Zhang  and Yueming Lu *

School of Cyber Space Security, Beijing University of Post and Telecommunication, Beijing 100876, China

* Correspondence: niezixiang@bupt.edu.cn (Z.N.); ymlu@bupt.edu.cn (Y.L.)

Abstract: The research topics of this paper are the data security of the edge devices and terminals of the Internet of Things (IoT) and the consensus design of a lightweight blockchain for the Internet of Things. These devices have self-organization capabilities to overcome the bandwidth delay and service-congestion problems caused by excessive concentration in existing scenarios, but they face the challenges of limited computing, storage, and communication resources. As a result, a non-financial lightweight blockchain consensus design with low energy consumption, low latency, and greater stability should be investigated. We propose a hierarchical proof-of-capability (HPoC) consensus mechanism combined with the asynchronous proof-of-work (PoW) mechanism for improving the computing capacity, storage capacity, and communication capacity of IoT edge devices that can generate blocks with low latency, low power consumption, and strong stability in resource-constrained edge device nodes, while ensuring that the security of the edge devices is enhanced asynchronously. We simulated a smart-home scenario, with the number of device nodes ranging from 15 to 75, and conducted comparative experiments between HPoC and PoW based on different difficulty bits. The experimental results showed that HPoC is a consensus mechanism with scalability and stability that can flexibly adjust time consumption and accurately select nodes with strong capabilities to generate blocks in heterogeneous devices.



Citation: Nie, Z.; Zhang, M.; Lu, Y. HPoC: A Lightweight Blockchain Consensus Design for the IoT. *Appl. Sci.* **2022**, *12*, 12866. <https://doi.org/10.3390/app122412866>

Academic Editors: Muhammad Zubair Asghar, Asad Masood and Shakeel Ahmad

Received: 21 November 2022

Accepted: 7 December 2022

Published: 14 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: blockchain consensus; lightweight blockchain; Internet of Things; proof of capability; asynchronous PoW

1. Introduction

The cost of integrated chips is falling as their size is shrinking due to quick upgrading and iteration. This has resulted in a rapid increase in the number of IoT nodes with considerable computational and storage capabilities, together with corresponding ideas, such as smart cities, smart factories, and smart homes, which have become hot topics in recent years in the field of IoT research. According to the GSMA, by 2025, there will be 1.8 billion mobile IoT connections worldwide (out of a total of 3.1 billion cellular IoT connections). Because of cost, volume, and power constraints, edge devices and terminals have limited computing, storage, and bandwidth resources, and they cannot guarantee online availability; thus, the standard server-client model cannot be applied to IoT nodes. Simultaneously, the transition from a closed centralized system to an open decentralized system has harmed the original system's excellent reliability, as small devices are more vulnerable to network assaults and a huge number of security flaws have been uncovered.

IoT edge devices, for example, can be readily managed to construct botnets [1]. The famous Mirai worm exploited IoT devices to launch large denial-of-service assaults [2]. The override-control and event-eavesdropping difficulties caused by access-control weaknesses in IoT cloud platforms are described in [3,4]. By exploiting communication protocol vulnerabilities in IoT devices, refs. [5,6] outlined denials of service, device hijacking, and replay attacks. The difficulties of system crashes, protection bypass, malicious commands, privacy leakage, and other issues caused by exploiting firmware vulnerabilities in IoT devices are addressed in [7,8].

IoT security and privacy issues are among the industry's most pressing concerns. Through the verification and consensus mechanisms of distributed nodes, blockchain technology [9], as the core technology of digital cryptocurrencies such as Bitcoin and Ethereum, can solve the problem of trust establishment between nodes of decentralized systems and realize decentralized, distributed trust-establishment mechanisms. Blockchain's transaction secrecy, security [10,11], data invariance, auditability [12,13], integrity, and fault tolerance [14] have led to its wide employment in many sectors outside of cryptocurrency. Blockchain is essentially a distributed data ledger that must maintain the global ledger's coherence, with the consensus mechanism of the blockchain constantly making this decision. For example, in PoW [15], the node with greater computational resources has a better chance of deciding the exact content of the next block, whereas in proof-of-stake (PoS) [16], the node with more tokens has a better chance of doing so. In a variety of proof-of-x consensus, nodes throughout the network must demonstrate that they are more capable in some way in order to become the leader in a particular consensus round and, accordingly, generate the next block. Because most existing blockchain systems are geared toward finance, they inevitably run into issues such as high power consumption and the need to store tokens, which are not suitable for fixing the security issues that plague IoT edge devices. In addition, when there are too many nodes, the design concept of nodes competing for block-generating rights across the network will drastically impair efficiency and increase network latency. An essential study topic for bringing blockchain to the IoT environment is how to allocate jobs of varying difficulty to nodes of various ability classes in a set of resource-constrained edge devices. In this article, we propose a lightweight hierarchical proof-of-capability consensus mechanism for IoT scenarios, based on both traditional proof-based consensus, such as PoW/PoS/proof-of-pure-stake (PPoS) [17], and the classical distributed protocol Raft [18]. Below, the important contributions of this study are set out in further detail:

- (1) In the IoT lightweight device scenario, we assess the existing conventional consensus and proof-based consensus mechanisms, show the related advantages and disadvantages, and analyze the demands and challenges of blockchain application in the IoT.
- (2) We propose a lightweight proof-of-capability (PoC) consensus mechanism that combines verifiable random function (VRF) and proof-of-storage and that can adapt to the resource-constrained IoT edge-computing environment. Furthermore, a double-leader election algorithm has been developed that can employ a pipeline mode for leader election and does not cause block-generation delay when the leadership node crashes.
- (3) We propose a heterogeneous hierarchical blockchain architecture that uses asynchronous PoW to offer proof of security. This design can transport block data to the cloud on time, transport clear data on edge nodes on time and adapt to edge-node storage capacity.
- (4) We model a smart-home scenario and conduct experiments on embedded IoT nodes to measure election time consumption and blockchain transactions per second (TPS), demonstrating the practicality of this design scheme. Comparative experiments based on PoW consensus were also carried out to show the superiority in stability, time consumption, and TPS.

The rest of this document is organized as follows: Section 2 discusses background knowledge about blockchain application to IoT security, blockchain consensus mechanisms, and lightweight blockchain. Section 3 mentions the problems that need to be solved and the corresponding solutions to implement a lightweight blockchain in IoT edge-computing scenarios. Section 4 discusses the design concept and execution procedure of the HPoC consensus mechanism, as well as the related block data structure. In Section 5, a smart-home scenario is created, with dozens of nodes, and an experiment comparing the HpoC and PoW consensus mechanisms is conducted. Finally, Section 6 provides a summary and evaluation of this paper's work, together with reflections on future work.

2. Background and Related Work

2.1. Applying Blockchain to Solve Security Problems in the IoT

The IoT has received greater attention and development in recent years than any other industry, due to its timeliness, interoperability, simplicity, and scalability, but the solution to its security challenges has not kept pace. By 2021, the cost of dealing with IoT information-security vulnerabilities rose to USD 119.9 billion. The features of blockchain technology, such as decentralization, self-organization, consensus mechanisms, smart contracts, and cryptography, are a natural fit with the IoT. In recent years, there has been a great deal of research to embed blockchain into the IoT to solve its information security issues. In general, blockchain can be applied in the IoT field to data privacy [19], data integrity, trusted accountability [20], trusted data-origin access control [21,22], and network information sharing [23].

As a step closer to applying blockchain to the IoT, some scholars have investigated how to design lightweight blockchain adaptively embedded into IoT edge devices. Ref. [24] combined two design algorithms, LSB and FogBus, then proposed a hierarchical and domain-specific design idea to reduce latency and resist attacks, adopting the leader idea of using a leader to process, verify, and distribute transactions. The leader node was named as a broker, but there was no detailed discussion on how a node becomes a broker; it was simply assumed that those who are capable can act as a broker. Ref. [25] used a lightweight encryption algorithm and stored medical privacy data in the cloud through a combination of symmetric and asymmetric encryption to provide data security and privacy for remote-patient-monitoring systems in the IoT attributes, but there was no discussion of the application of blockchain itself in this IoT environment. Ref. [26] provided an idea of optimizing the storage volume from the ledger structure to reduce the storage pressure of common node injuries.

Accordingly, research on lightweight blockchain is still in its infancy. The available work is limited, but can be broadly divided into the following three points:

- (1) Partitioning or sub-domaining a blockchain to form numerous chains in parallel, with the ability to interact between parallel blockchains.
- (2) Separating the on-chain and off-chain storage of blockchain data to relieve the data pressure on the blockchain, or partitioning the data storage in Merkle Tree style and requesting data from other nodes when verification is required.
- (3) Dividing the roles of nodes in the blockchain and nominating more capable nodes to be responsible for more computation and storage work, so as to avoid all nodes participating in the block-generation or computation work.

In short, the existing works describe the lightness of a blockchain in a broader sense, as if it were a plug-and-play module in the context of IoT applications. If an application is more granular, different blockchain types are discussed, such as using a public chain, a federated chain, or a private chain. Blockchain systems that are tailor-made for IoT edge-computing devices are unusual for this type of work. Instead of considering a lightweight concept in the true sense, some researchers believe that combining blockchain technology with the Internet of Things could simplify some of the security-authentication processes, thus achieving the goal of lightweight device security. In this work, we argue that running blockchain on IoT nodes will raise the computational load on these resource-constrained edge devices, especially when employing PoW and other consensus techniques that only involve mining, which is counter to the goal of lightweight devices. As a result, lightweight features should be used to describe the nature of blockchain, such as the time consumed by the blockchain consensus process and the data storage load in the blockchain. Accordingly, in this paper, a lightweight blockchain consensus and architecture design for IoT scenarios, by combining the advantages of various existing consensus mechanisms, is proposed.

2.2. Development of Blockchain Consensus

The problem of consensus mechanisms originated from the distributed database consistency problem in the 1970s and 1980s, and [27] proposed the Two Armies Problem in

the computer field. Refs. [28,29] proposed the famous Byzantine general problem and the solution algorithm Paxos. Ref. [30] designed the first practical Byzantine problem solution algorithm. Until 2008, the research on consensus mechanisms was limited to the small-scale node database consistency problem; ref. [9] applied a consensus algorithm in a massive-scale open-node network and provided it with a financial currency property, and the research into consensus algorithms has ushered in a rapid development period during the last decade. Successively, PoW, PoS, PoC, DPoS [31], PoE [32], and PoUS [33] were generated. In general, the consensus applied in the existing blockchain technology can be broadly classified into classical consensus and proof-based consensus. Both types of consensus have their advantages and disadvantages: the advantage of classical consensus is that it is fast, but the disadvantages are that it is not fair enough and there is no discussion on the choice of leadership nodes. The advantages of proof-based consensus are that it is decentralized and more democratic, but it consumes corresponding time, power, and financial resources, depending on its type. Thus, a hybrid consensus scheme combining the two kinds of consensus has become a new research direction in recent years. Most of this hybrid consensus uses proof-based consensus to select a certain number of leading nodes from the whole-network nodes, then uses classical consensus to reach a consensus on the block and, finally, broadcasts to the whole-network nodes. For example, ref. [34] proposed improving the performance of the PoW protocol by decoupling blockchain operation into two planes: leader election and transaction serialization. Compared with traditional PoS consensus protocols, the BFT-style PoS provides a deterministic finality to guarantee that finalized blocks cannot be revoked. Furthermore, deterministic finality allows for designing a reward-and-punishment strategy to discourage malicious validators from launching attacks, such as double-betting or nothing-at-stake attacks [35,36]. However, integrating cryptocurrency-oriented blockchain technologies into IoT systems meets tremendous challenges of scalability, storage capacity, security, and privacy. Given the aforementioned difficulties in integrating blockchain technology into IoT systems, building an optimized blockchain with light and efficient consensus algorithms appears to be a prospective solution.

3. Problem and Solution Analysis

According to the background work described in Section 2, blockchain consensus is progressing toward hybrid consensus, which combines the fairness and unpredictability of proof-based consensus with the non-financial properties of classical consensus. When applied to the IoT situation, the hybrid consensus must take into account the issues set out below.

3.1. Stages of Hybrid Consensus

The majority of current work on integrating lightweight blockchain into IoT-scenario applications fails to grasp that hybrid consensus is a multi-stage process that includes both the election and block-generating stages. Hybrid consensus often requires a leader or a leadership committee, and the election stage will determine which nodes could fill these roles; after the election is complete, it will move to the block-generating stage, where the network-wide nodes will accept and endorse the blocks generated by that leader node or leadership committee. The election stage and the block-generating stage are two distinct stages that should be discussed individually, although there may be some crossover. The intermittency of blockchain creation is also affected by whether the election and block-generating stages are parallel or serial. For example, in the serial form of transition between the election stage and the block-generating stage, when the election stage is in progress, the block-generating stage of the blockchain will start to pause and no new blocks will be generated until a new leader node is elected.

3.2. Election Logic and Randomness in Line with the IoT Scenario

The IoT scenario is at odds with financial scenarios such as Bitcoin and Ethereum. Nodes do not spend a great deal of computing power competing for block-generation rights, and there are no tokens that serve as collateral. Since users are motivated to operate the blockchain for security reasons, financial incentives are not appropriate at this time. IoT edge devices consist of a large number of heterogeneous embedded terminals, such as smartphones, sensing devices, smart appliances, personal computers, routers, and gateways. In essence, devices with stronger computational, storage, and bandwidth resources are more suitable to serve as leader nodes that are responsible for block generation. Therefore, the logic of the election should efficiently consider the above three resources together as a capability that can act as a leader node. Furthermore, because these devices are inherently unstable, the blockchain should have sufficient robustness to replace the leader node at any time.

3.3. Reducing Storage Pressure on Normal Nodes

The blockchain is a time-series database, with data volume increasing in lockstep with chain length. Non-leader nodes frequently have insufficient storage capacity to keep all of the block body data and must periodically purge their stored block data. For instance, once a certain period of time has passed, only block headers (which include Merkle Roots capable of verifying block body data) will be kept, while a substantial amount of block body data will be destroyed or moved to a cloud with more powerful storage capacity. This reduces the storage burden of non-leader nodes and assures that they can validate subsequent block contents. It is also worth considering whether the data structure needs to be altered after the data are regularly sent to the cloud for storage.

3.4. Proof of Security and Tamper Resistance of Blockchain

A blockchain is a data structure made up of a series of blocks that are linked together by a cryptographic hashing algorithm. Each block's header contains a hash, which may be thought of as a collection of all the information from the previous block. Because hashing is collision-resistant and one-directional, it allows users to check the consistency of blocks right away. However, the blockchain is not impeccably secure; by changing the hash in the block header sequentially from one block to the next, a new chain with an entirely different chain can be created, a phenomenon known as forking in blockchain systems. The PoW is based on obtaining a random number that matches the criterion, and only nodes that have paid a high computational cost may do so. This makes rebuilding a new chain prohibitively expensive, thus preventing malicious tampering. While the process of locating that random number provides the blockchain with tamper resistance and security, it also drastically slows down block-generating efficiency. It might be claimed that PoW is a two-edged sword that provides security while reducing efficiency, and that a method is needed to offset this impact while still contributing to the block's security.

To summarize, developing a lightweight blockchain in the IoT edge-computing scenario necessitates not only a holistic approach and a novel design from the standpoint of consensus, but also a unique ledger layout and data structure. We explore the four issues raised above in this section and provide remedies, such as constructing a double-leader parallel election model to reduce the election stage's impact on the block-generating stage. An election algorithm based on PoC combined with VRF is designed to ensure the randomness of the election. In addition, a hierarchical structure consists of minute blocks, ten-minute blocks, and day blocks with which asynchronous PoW is used, which can reduce the storage pressure on non-leader nodes and separate the PoW computation process from the block-generating process to ensure the blockchain's security and tamper evidence while maintaining a high block-generating speed. The next section will discuss the specific consensus process and algorithm design, as well as the corresponding data structure and chain architecture.

The proposed method has the following advantages: (1) low price and ultra-low power consumption—POC only needs a computer hard disk to mine, so the cost of mining input is lower; (2) the refusal to monopolize, with stable income—POC is relatively fair and naturally anti-monopoly; (3) strong risk resistance—POC is easier to sell or be reused.

4. Consensus Design and Chain Architecture

The consensus based on hierarchical proof-of-capability is divided into two parts: a hybrid consensus design that incorporates both classical and proof-based consensus design concepts, and a hierarchical chain-architecture design. Across the hybrid consensus, a LEADER node is elected for the entire network, and the LEADER node has the concept of a TERM in classical consensus, which the LEADER generates and broadcasts to all blocks within a fixed period. At the same time, in order to avoid the blockchain being shut down during the LEADER changeover, this article proposes a double-leader paradigm, in which each round of the TERM has a LEADER and a VICE-LEADER. When the VICE-LEADER node drops out, the LEADER conducts the election of the new VICE-LEADER and the next TERM begins. When the LEADER node fails, the VICE-LEADER takes over as the new LEADER, then elects a new VICE-LEADER, and the next TERM begins. After the genesis block (the first block) is formed, the process of electing and generating blocks runs in parallel. The hierarchical chain structure can greatly reduce the storage pressure of ordinary nodes and provide a channel for subsequent security enhancement. The election's PoC algorithm, the block's data structure, the related consensus flowchart, the hierarchical chain architecture, and the asynchronous PoW mechanism that offers security are all detailed below.

5. Election Algorithm Based on Proof-of-Capability

The capabilities of an IoT edge device are attributed in this study to storage resources, computer resources, and bandwidth resources. The stronger the capability, the more probable it is that the device will become the LEADER node for the entire network.

The nodes contending for block-generating powers in the PoW consensus must suggest a random number that meets the criteria that the hash value computed from that random number is small enough (i.e., the number of prefix bits reaches the number of zeros that meet the requirement). As a result, the range of this random number has no limit as long as the computation result fulfills the criterion, making the search for this hash riddle borderless, causing a fierce rivalry for computing resources and a large amount of power waste.

In this paper, the hash function is used to solve the concept of the random number, but the boundary of the random number is limited. In each round of elections, random integers needed to solve the hash problem are selected at random from the largest cached data table. The idea for this boundary comes from the storage proof consensus, which dictates that each node maintains an affordable table in its own chip cache and that the cached table held by a large-storage capacity node must include the cached table held by a small-storage capacity node. The ELECTION HASH for broadcasting to the nodes of the entire network is obtained after a specified calculation process. The ELECTION HASH is sent to each node, which then iterates through the cached data tables it keeps to find the corresponding random number. The ELECTION HASH can be computed by nodes that have cached the random number, but nodes that have not cached the random number are unable to find a random number that fits the conditions. Nodes with larger data tables are theoretically more likely to locate the answer, because the selection of random integers is a random process. Nodes with the same cache capacity must still perform the iterative calculation procedure to determine the random number. The stronger the node's computational power, the faster the computation, and this phase filters the node's computational resources.

Simultaneously, the node that discovers the random number must broadcast its own ELECTION PROOF to other nodes and gather VOTES from them. Because each node can only vote once every election round, nodes with better network conditions can broad-

cast their ELECTION PROOF to other nodes faster, filtering bandwidth resources. The idea behind this consensus is to use small-scale computation for capability proofs in a restricted range of random integers, which may realistically distinguish gaps in storage, computational, and bandwidth resources among edge device nodes while keeping energy consumption and latency within acceptable limits.

ELECTION HASH is an important concept in this paper, and because the ELECTION HASH is computed from a random number by means of a hash function, the number is restricted to a cache data table. The question of whether the random number and the ELECTION HASH can be predicted and calculated in advance must be considered, and this work employs the notion of RANDOM SEED to achieve unpredictability and proved randomness of the random number and the ELECTION HASH. The idea of RANDOM SEED comes from the VRF, which was proposed in 1999 [37]. As shown in Equations (1)–(3), for a particular input m and the inputter's private key Sk , the VRF outputs a RANDOM SEED s and a VRF proof, and the verifier can verify by the output RANDOM SEED s , the proof, the inputter's public key Pk , and the input m that the RANDOM SEED s is generated by this input m . This process is secure, as it does not need to expose the private key Sk of the inputter.

$$s = \text{VrfHash}(Sk, m) \quad (1)$$

$$\text{proof} = \text{VrfProof}(Sk, m) \quad (2)$$

$$\text{True/False} = \text{VrfVerify}(Pk, m, \text{proof}, s) \quad (3)$$

VRF has been used in many blockchain consensus, such as [18,38,39]. In this paper, the RANDOM SEED is generated by the VRF and is hashed after stitching with a random number to obtain the ELECTION HASH.

Because PoC uses a double-leader model, two elections will be conducted during the consensus startup phase; the LEADER node will be elected first, and then the VICE-LEADER node will be elected. After that, only the election of the VICE-LEADER will be conducted during the consensus running phase (by default, the case in which both the LEADER and the VICE-LEADER drop out at the same time is not considered). The algorithm uses a node's private key and the current timestamp to construct a VRF RANDOM SEED, as well as the VRF proof to verify it. After that, a random x -bit random number (an x -bit is the maximum number of bits in the cache data table of the whole network, which means the random number is selected in the range of 2^x) and the RANDOM SEED are stitched together to obtain the ELECTION HASH. The ELECTION HASH and the RANDOM SEED, VRF proof, and public key of this node are packaged as ELECTION DATA and disseminated to the whole network. The pseudo code of Algorithm 1 is as follows:

Algorithm 1 Election Hash Generation

```

NodeP = RandomSelect (Nodegroup)
randomSeed = VrfHash (Skp, TimeStamp)
proof = VrfProof (Skp, TimeStamp)
randomNumber = RandomSelect (2x)
elecHash = Hash (randomSeed + randomNumber)
elecData = Package (elecHash, Pkp, randomSeed, TimeStamp, proof)
Broadcast (elecData)

```

Algorithm 1 describes a process in which node P generates and broadcasts the ELECTION HASH. Then, all nodes will run the leader election algorithm after receiving the ELECTION DATA. The validity of the RANDOM SEED is first checked, and then they proceed to calculate the ELECTION HASH by traversing the cache data table, which they manage themselves (traversing the selected data and RANDOM SEED are stitched together to perform the hash calculation, and then compared with the ELECTION HASH in the ELECTION DATA). The node that discovers the random number signs it before broadcast-

ing the random number, signature, and public key to the whole network. After receiving the packet, all nodes in the network will check it, and if the verification is successful, they will vote for the node and append their signatures. In each term, each node can only vote once. Algorithm 2 depicts how a Node q participates in the election of the leader.

Algorithm 2 Leader Election

```

Receive (elecData)
If True VRFVerify (Pkp, randomSeed, TimeStamp, proof) For x in CacheTable:
If True Equare (Hash(x + randomSeed), elecHash) Signature = Sign (Skq, x)
elecProof = Package (Signature, Pkq, x) Broadcast (elecProof)
ReceiveVotes()

```

The LEADER node is the first node to collect more than half of the network's voting signatures and to begin generating the genesis block, which records the TERM and the LEADER node's metadata. The leader node will then immediately begin the VICE-LEADER election, which is identical to the LEADER node election except that the node that performs Algorithm 1 becomes the LEADER node. The election procedure is the same in the ensuing time-of-consensus operation.

Figure 1 illustrates the process of PoC consensus. At the beginning, a node selected from the whole network chooses a random number, executes the algorithm to calculate the ELECTION HASH, and generates the corresponding VRF information. After that, the ELECTION information will be broadcast to the whole network. All nodes first verify the correctness and legitimacy of the ELECTION information received. After the verification is passed, the nodes start to participate in the LEADER election. The first node to find the corresponding random number (assuming node A) signs and packages the relevant information into the ELECTION PROOF and broadcasts it. Other nodes also verify after receiving the ELECTION PROOF, and vote for node A after passing the verification. When node A collects more than half of the votes of the whole network, it automatically becomes a LEADER node and starts its first TERM. After the TERM start, node A would collect Tx and start the election of VICE-LEADER. simultaneously. After the same election process, it is assumed that node B becomes VICE-LEADER. Node A and Node B will start the heartbeat monitoring mode. Once one of the nodes goes offline, the other node will start the next TERM and elect a new VICE-LEADER. When both LEADER and VICE-LEADER are online normally, the TERM remains unchanged and the blockchain network operates normally.

5.1. Data Structure of Block

The data structure of the block, particularly the data structure of the block header, is strongly associated with the design concept of blockchain. Except for the consensus startup phase, nodes interact in PoC via the block header, which contains information such as the TERM, ELECTION HASH, RANDOM SEED, VRF proof, information about the LEADER node and the VICE-LEADER node, and so on. The non-LEADER nodes obtain the block broadcast by the LEADER node and examine the block header content. If the TERM field in the block header changes, it indicates that a new round of voting has begun and then the fields that will begin voting for the new VICE-LEADER are identified. The block's data structure is depicted in Figure 2.

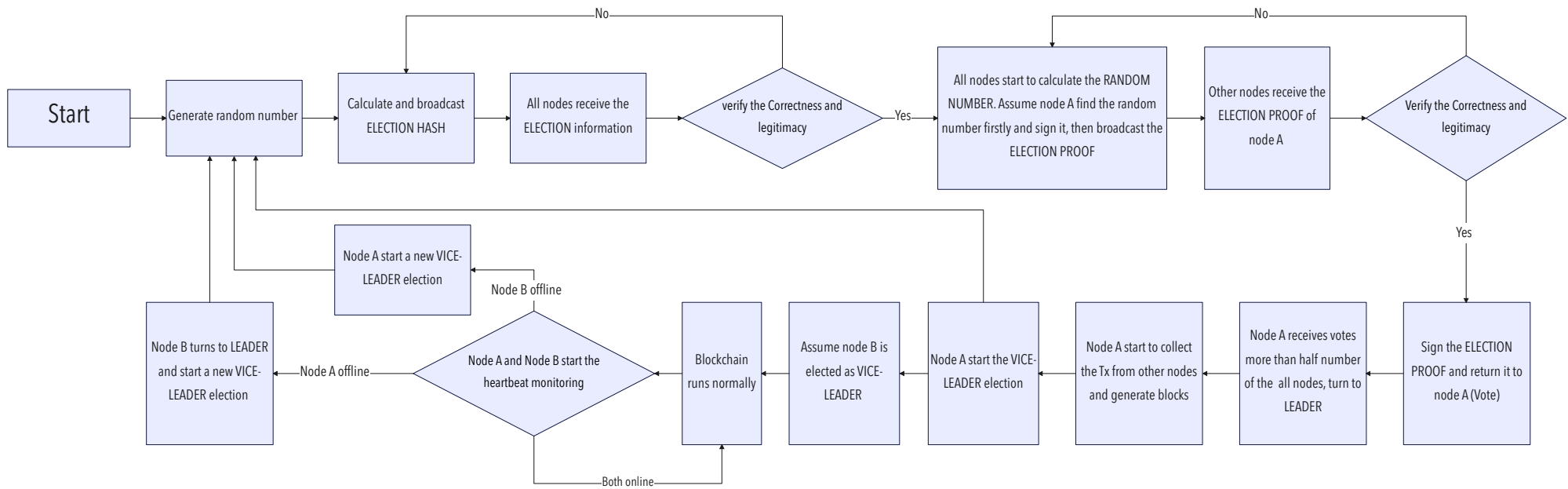


Figure 1. An illustration of PoC consensus process.

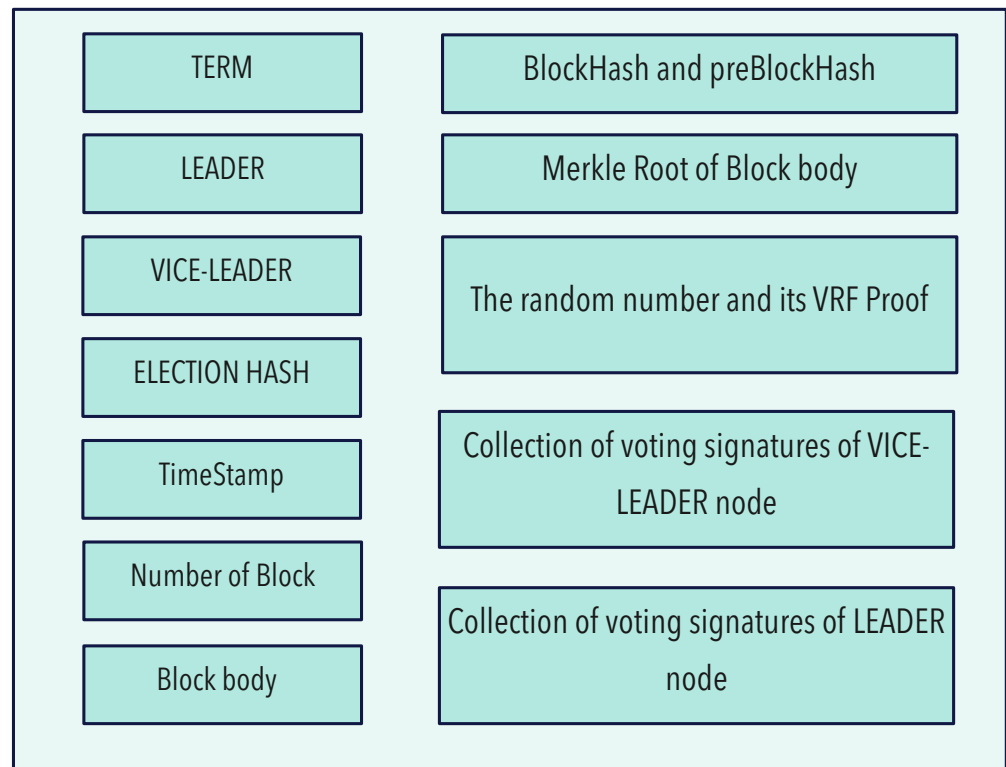


Figure 2. Data structure of the block.

It should be noted that some block header fields may be null. When the VICE-LEADER, for example, drops the line, the LEADER will begin the next TERM and elect a new VICE-LEADER. However, because the LEADER is now creating blocks in parallel, the VICE-LEADER field in consecutive blocks will be null until a new VICE-LEADER is elected and its information is logged in the most recent block. The Timestamp, Term, Number of Blocks, Block Hash, and Parent Block Hash are never empty, allowing each block to be identified.

When a node finds the random number corresponding to the ELECTION HASH, it will sign it and package it with its own public key, and then broadcast the data packet to the whole network. When other nodes receive the packet, they will conduct a verification. If the ELECTION HASH corresponding to the random number is correct, they will vote and sign the node, and package it with their own public keys and return it to the node that finds the random number. Each node can only vote once in each round of election, and once it has voted, it will no longer participate in the election; i.e., it will stop looking for the random number corresponding to the ELECTION HASH.

5.2. Consensus Flow Chart

A six-node network is provided in this section to demonstrate the various stages of the consensus. First, some roles, data flows, and concepts are introduced. For better understanding, the demonstration method is kept as simple as feasible. The flow charts will be explained next to demonstrate the consensus.

The roles are as follows:

- **LEADER:** the node in charge of gathering transaction data from other nodes around the network, manufacturing blocks (including minute and ten-minute enhancement blocks with proof of work), and delivering the enhancement blocks to the cloud server. It is either formed during the initial round of voting or converted by succeeding VICE-LEADER nodes.
- **VICE-LEADER:** the standby node of the LEADER node. Heartbeat monitoring is conducted between the LEADER node and the VICE-LEADER node. When the leader

drops off, the VICE-LEADER node will be automatically converted to a new LEADER node and initiate a new TERM and an election of a new VICE-LEADER.

- CANDIDATE: the node eligible for election (the node that can find the random number corresponding to the ELECTION HASH in its local digital cache table), which is usually a node with abundant computing resources and storage resources.
- FOLLOWER: the node that is only responsible for generating the transaction of the blockchain. The computing resources and storage resources of FOLLOWER nodes are relatively scarce; they only store the block data within a certain period of time and clear it regularly.

Data streams:

- Election proof: a data packet that is signed and broadcast to other nodes after the random number corresponding to the ELECTION HASH is calculated by the CANDIDATE nodes.
- Vote information: after receiving the Election proof from a Candidate node, the other nodes will verify the legitimacy of the message, i.e., hash verification and signature verification. After the verification is passed, the message is signed and the signature is returned to the corresponding CANDIDATE node. Each node can only vote once for each election round.
- Vote information collection: after CANDIDATE nodes receive vote information from more than half of the nodes in the network, the vote information will be packaged as their Vote information collection.
- Tx data: the data information generated by the IoT terminals are composed into transaction data that make up the blocks through a fixed format. All nodes will generate Tx data.

Other concepts:

- Election: the process of generating the LEADER node and the VICE-LEADER node. The LEADER election will be executed only in the first round of election, and subsequent elections will only execute VICE-LEADER election (the subsequent LEADER node is converted from the VICE-LEADER node). All subsequent elections, with the exception of the first round, are run in parallel with blockchain generation.
- Term: a fixed period of time for the LEADER node to work. Each round of a term has a pair composed of LEADER and VICE-LEADER; the term will not change until the LEADER or VICE-LEADER is offline. Meanwhile, the commencement of the next round of elections (VICE-LEADER election) is marked by the change of term, and the term information may be found in the block header. When a non-LEADER node sees the change of term in the block header, it looks at the election information in the block header, such as ELECTION HASH and RANDOM SEED, and decides to vote.
- Heartbeat monitoring: the mutual listening mechanism between the LEADER node and the VICE-LEADER node.

To ensure that the election and block generation are parallel, when the VICE-LEADER node fails, the LEADER node will open the next term and the election of a new VICE-LEADER; when the LEADER drops out, the VICE-LEADER automatically becomes the LEADER and opens the next term and the election of a new VICE-LEADER. Because LEADER and VICE-LEADER are unrelated nodes located far apart, the chances of both being offline at the same time are extremely remote; hence, this study does not investigate the possibility of LEADER and VICE-LEADER going offline at the same time.

- Digital cache table: each node in the network maintains a digital cache table in RAM according to its storage capacity, and the numbers used to calculate the campaign hash are randomly generated from the specified maximum cache. This means that the node that stores a larger cache table is more likely to find the random number corresponding to the campaign hash in each election round, while the node that maintains a cache table of the same size has more computational power to find the number corresponding to the ELECTION HASH faster. As a result, nodes with larger

storage and computing resources are more likely to become leader nodes, which is compatible with the capability proof concept presented in this study. The digital cache table displayed in the figures is intentionally kept modest for demonstration purposes (only 10 numbers are generally stored), although the table is a massive data table in reality.

- Offline: this refers to the case where the LEADER node or the VICE-LEADER node loses the ability to provide services.
- Blocks: blockchains generated by LEADER nodes; in this case, blocks specifically means minute blocks and ten-minute enhanced blocks.

In Figure 3a, the whole-network nodes receive the ELECTION HASH and start to look for the corresponding random number. Assuming that the random number is 7, only node A and node B can find the random number, so they become CANDIDATE nodes. As node A has stronger computing power, it finds the number 7 first and sends the election proof to other nodes. Other nodes will return the vote information to node A after receiving the election proof and passing the verification. Once node A receives the vote information with more than half of the number of nodes in the whole network, it considers itself a LEADER node. In Figure 3b, at the beginning of TERM 1, node A becomes the LEADER node, generates the genesis block, and starts the election of the VICE-LEADER node. This time, assume the random number is 6. Both node B and node E have the conditions to become a CANDIDATE node. Because node B has stronger computing power, it is the first to find the random number 6 and starts sending the election proof to the whole network and collecting the corresponding vote information. When node B collects enough vote information, it will send it to the LEADER node (node A). After receiving it, node A will verify it. After passing the verification, node B will be designated as a VICE-LEADER node.

Figure 4a shows the consensus period when the double-leaders are online and the whole network operates normally. That is, all nodes submit block TX data to node A, which collects the data and produces blocks before broadcasting to the entire network. In addition, nodes A and B will check each other's heartbeats to see if the other is still connected. When the VICE-LEADER drops the line, the LEADER node (node A) starts the next TERM and elects a new VICE-LEADER, as shown in Figure 4b. Nodes C, D, and E have become candidate nodes in this round of voting, and node E is the first to find the random number and broadcast to the entire network.

Figure 5a shows that node E has become the VICE-LEADER node in TERM 2 and has executed the heartbeat monitoring with node A, and the whole network has entered the normal working stage again. Figure 5b shows that node B is back online. However, in TERM 2, it has become a FOLLOWER node and is only responsible for generating TX data.

Figure 6a illustrates that when node A (LEADER node) is offline, node E will automatically become a new LEADER node and start the next TERM, and publish the ELECTION HASH of the new VICE-LEADER in the block with a new TERM. Because node B has stronger storage and computing power, it is the first to find the random number corresponding to this round of elections. Figure 6b shows that node B has become the VICE-LEADER node in TERM 3, and has opened the heartbeat monitoring with node E. At the same time, node A is back online and becomes a FOLLOWER node. The whole network enters the normal working stage again.

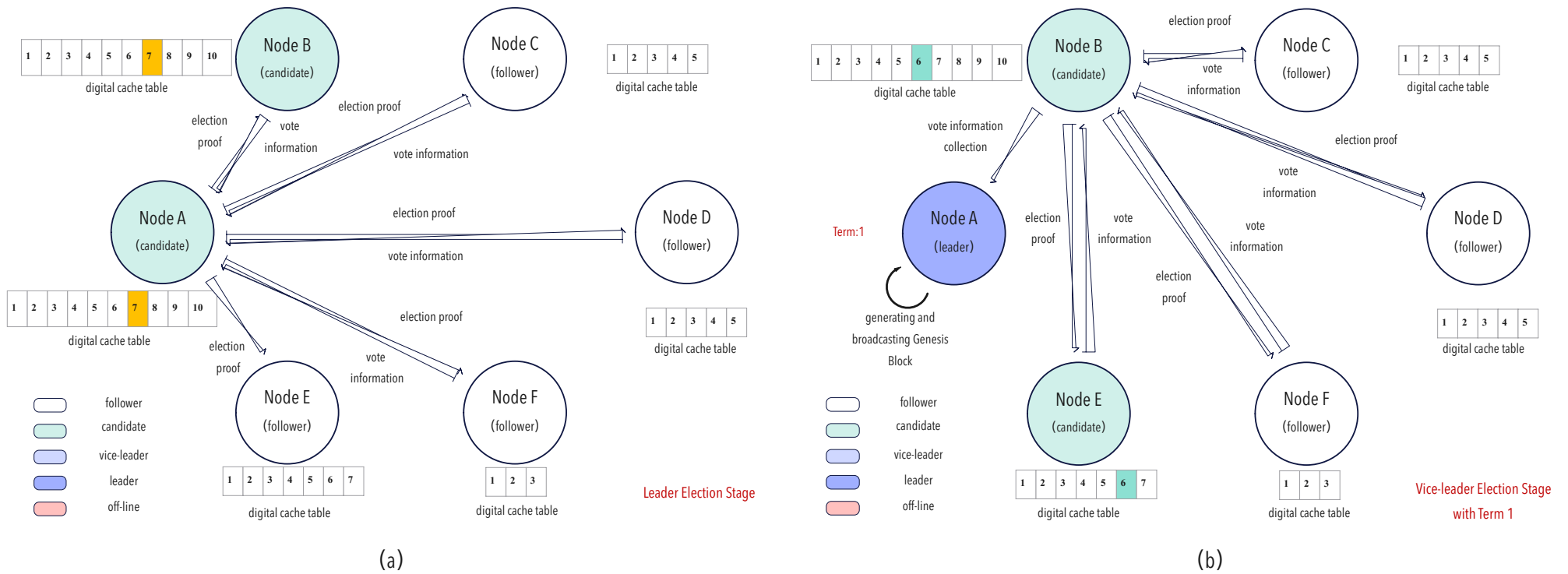


Figure 3. Consensus flow chart: (a) LEADER election stage and (b) VICE-LEADER election stage with TERM 1.

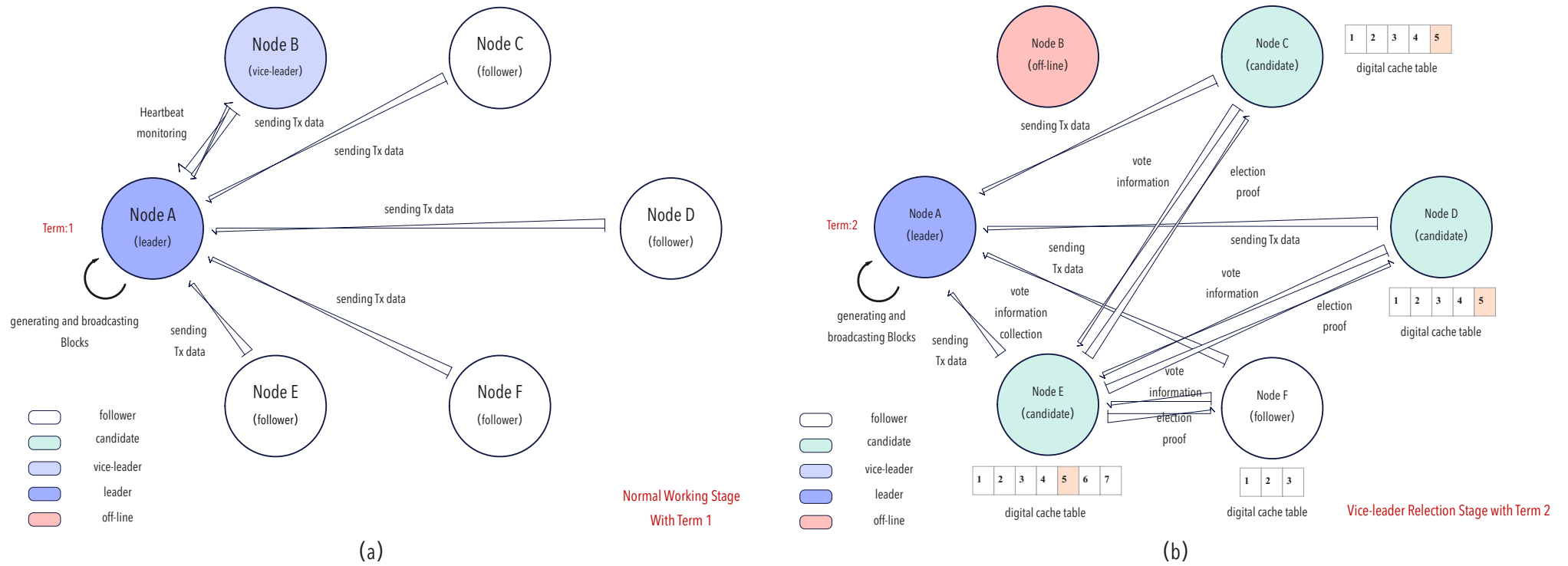


Figure 4. Consensus flow chart: (a) normal working stage with TERM 1; (b) VICE-LEADER reelection stage with TERM 2.

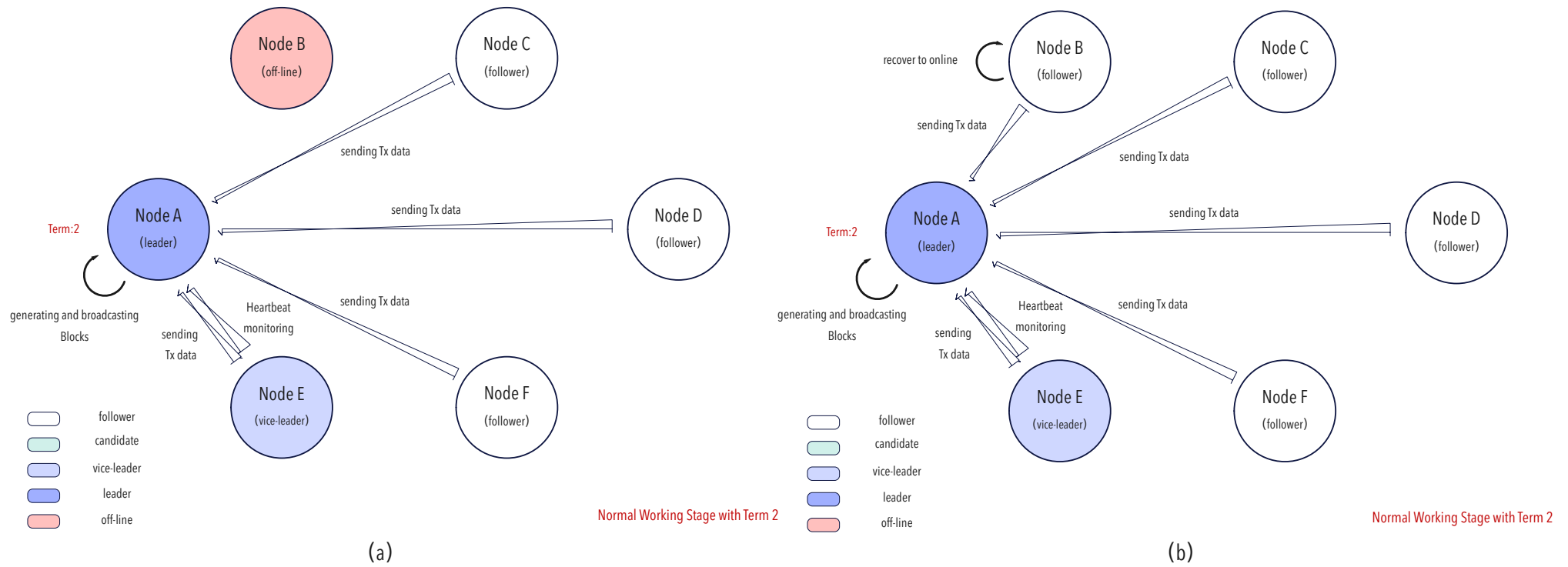


Figure 5. Consensus flow chart: (a) normal working stage with TERM 2 (Node B offline); (b) normal working stage with TERM 2.

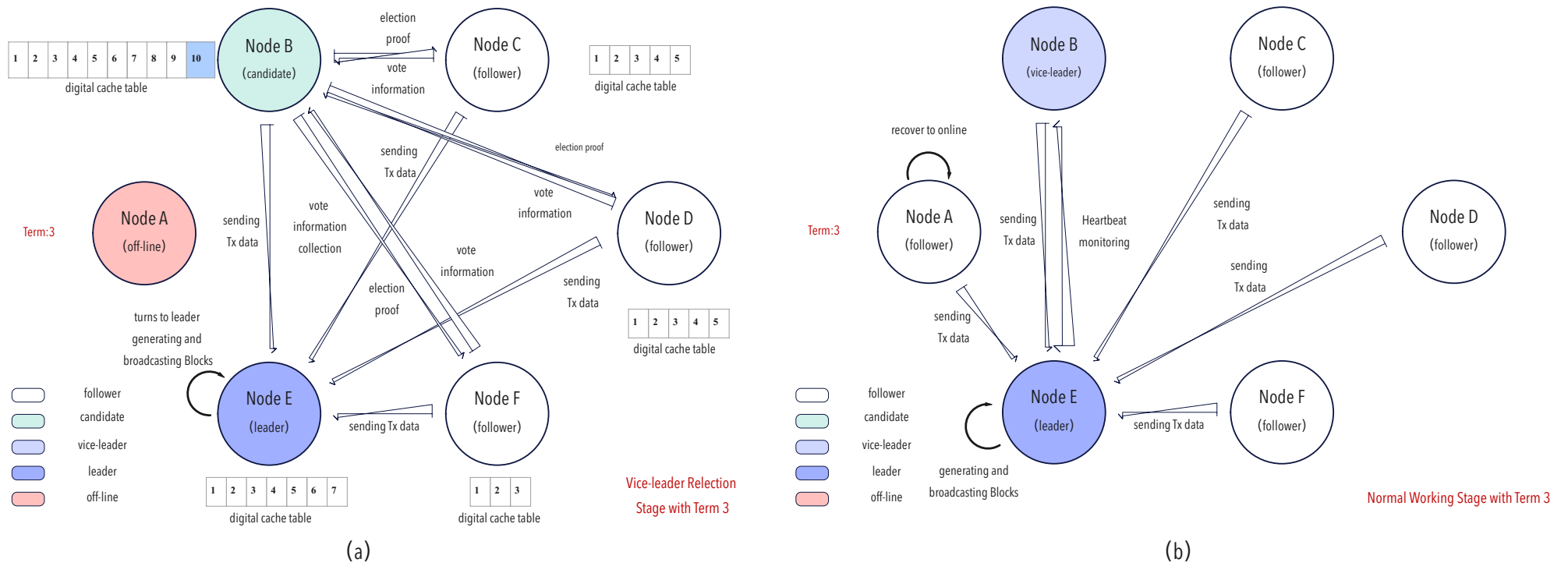


Figure 6. Consensus flow chart: (a) VICE-LEADER reelection stage with TERM 3; (b) normal working stage with TERM 3.

5.3. Hierarchical Chain Architecture and Asynchronous PoW
 Hierarchical Chain Architecture: To Decrease the Data

As a burden on blockchain nodes, the hierarchical chain-architecture design is widely employed. In this study, we propose a hierarchical chain architecture that includes a minute chain, a ten-minute chain, and a day chain, with blocks generated every minute, every ten minutes, and every day, respectively. The non-transaction contents of the blocks, such as the block number and the date, are prefabricated. The LEADER node that generates the block merely aggregates the transactions within that minute and packs them into a premade block for minute blocks. The minute blocks are saved in the LEADER node and the FOLLOWER nodes, but only the minute blocks from the previous seven days are stored on the nodes to decrease the storage burden on the normal nodes, and the minute blocks from earlier than that period are automatically purged. Every ten minutes, the LEADER node merges and packs the minute blocks in order to make a ten-minute block, a type of block with better security for minute blocks. The LEADER node keeps the ten-minute blocks for a long period before uploading them to a cloud server with more computing and storage power, which will combine and construct a daily block, a type of block with increased security for ten-minute blocks. The design of this hierarchical chain architecture is shown in Figure 7.

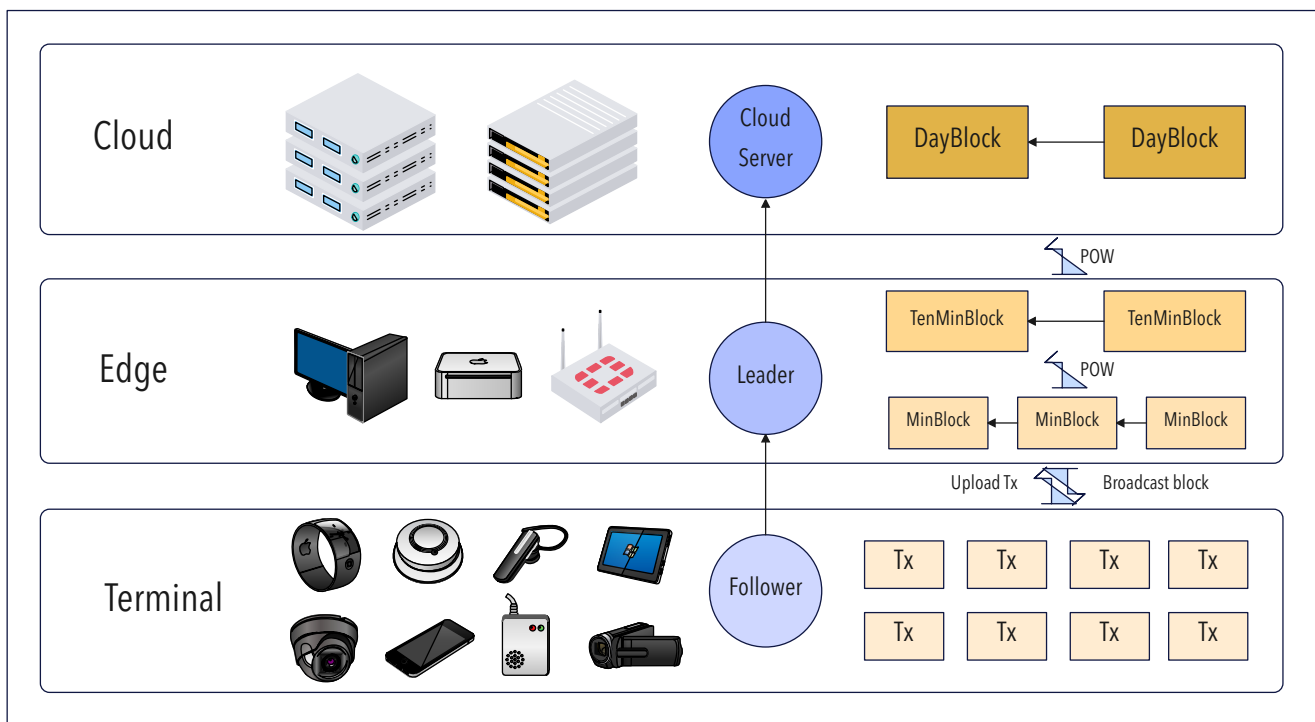


Figure 7. Hierarchical chain architecture and asynchronous PoW.

Asynchronous PoW: The ten-minute block enhances the security of the minute-block, and the day block enhances the security of the ten-minute block. Asynchronous PoW is employed in this architecture, which means that PoW is performed exclusively in the ten-minute and daily blocks, with the process having no effect on the production of the minute-blocks. The tamper-proofness of the block is continually increased as the security of the blockchain is continuously enhanced asynchronously in the followup. The difficulty bits of PoW will be modified in real time, based on the LEADER node’s power and computational load as well as the cloud server’s. The asynchronous PoW mechanism separates the node’s mining and block-generating processes, and minute blocks are issued once every minute, resulting in 1440 min blocks every day, with a PoW time interval of 10 min and one day.

This can increase the blockchain transaction volume while also ensuring security (see Table 1).

Table 1. Experiment set up of hardware and software.

Experiment Setup	Description
CPU	Phytium, FT-2000+/64
RAM	128 GB
HDD	4 TB
System	KylinOS

6. Comparative Experiment and Simulation Results

This section simulates a smart-home scenario consisting of heterogeneous computing device nodes and implements an HPoC consensus-based blockchain. Nodes are classified into five classes and their storage and computational capacities decrease from Class 1 to Class 5 in order. The storage capacity is simulated by the size of the digital cache table, while the computational capacity is simulated by the random waiting time to begin calculating the random number (from 100 ms to 500 ms). The number of nodes is divided into groups of 15 the ratio of the number of nodes in one group from Class 1 to Class 5 = 1:2:3:4:5. A total of one to five groups of experimental statistics are conducted and the total number of nodes gradually increases from 15 to 75. In addition, this section implements a comparison experiment of the PoW consensus process to show HPoC's superiority in terms of stability of consensus process-time consumption. The data for the experiments are averages obtained from the data of twenty consensus rounds. All programs are developed based on open-source technologies, such as go language, TDengine, and ETCD. The experiment setup information of hardware and software is shown in Table 1.

The difficulty bits in the HPoC consensus reflect the size of the digital cache table kept by the Class 1 nodes (the nodes with the largest storage capacity) and the number stored in the cache table varies from 0 to the 2^{25} . Class 2 nodes have a 24-bit cache table, Class 3 has a 23-bit cache table, and so on. When the difficulty bits are set to 25, the time to complete a consensus round in the network is between 45 and 70 s, as illustrated in Figure 8a. The number of nodes has no effect on the consensus time consumption, which fluctuates as the number of nodes increases (showing that the consensus time consumption is mostly determined by the random number rather than by the number of nodes); its typical duration is about 60 s. When the difficulty bits are set to 24, the average consensus time consumption drops to 30 s, and when the difficulty bits are set to 23, the average consensus time consumption drops to roughly 15 s. This shows that the network-wide consensus consumption time is halved for each decrease in difficulty bits, and all four difficulty bits show the scalability advantage of the HPoC consensus, with the number of nodes having no effect on the network consensus consumption time when the number of nodes is increased from 15 to 75.

For three difficulty bits, we compared the coefficient of variation of time consumption via HPoC and PoW in Figure 8b. Because HPoC is built as a lightweight consensus protocol, its difficulty bits must tend to be selected as smaller in the process of practical application in order to meet the goal of reducing time and power consumption; we chose not to directly compare the two consensus time consumptions with the same difficulty bits. Furthermore, the coefficient of variation can be used to determine the degree of dispersion of a group of values, or the extent of the deviation from the average. When the difficulty bits remain the same, the coefficient of variation of HPoC stays around 0.5, whereas the coefficient of variation of PoW stays around 1 or even up to 1.5. This means that the PoW consensus has a low level of stability, and obtaining consensus can take anywhere from a few seconds to a hundred seconds.

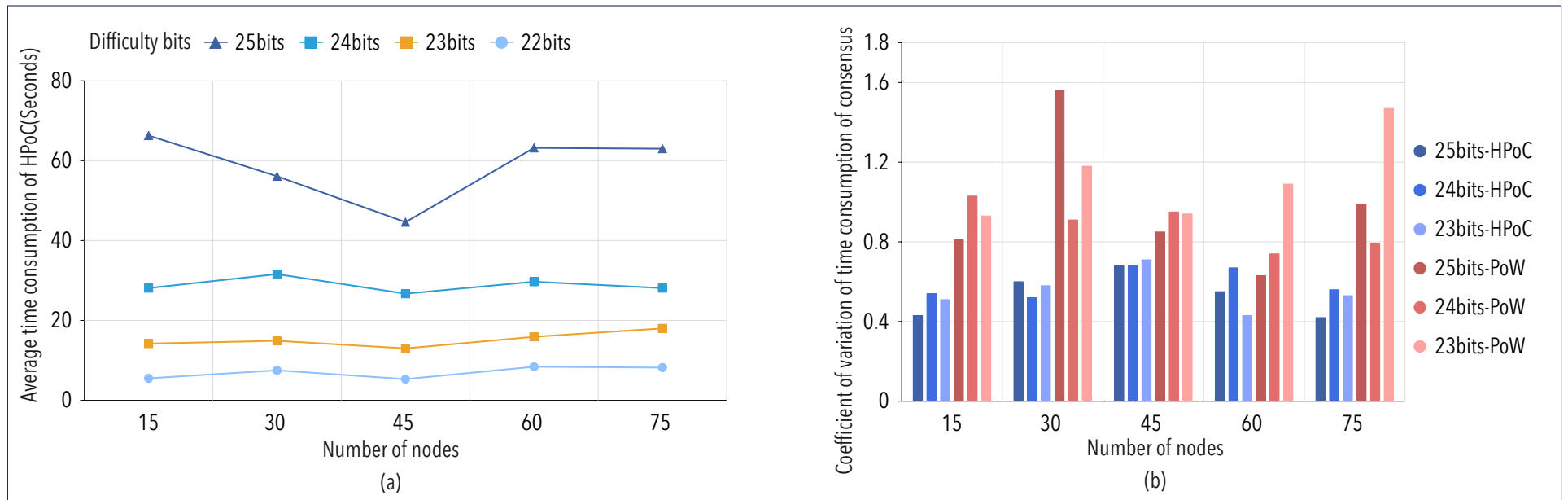


Figure 8. Simulation results: (a) average time consumption of HPoC with four difficulty bits; (b) comparison of coefficient of variation of time consumption between HPoC and PoW with three difficulty bits.

The IoT environment needs a stable consensus mechanism for blockchain services, and a consensus with a smaller dispersion coefficient is more capable of meeting this requirement.

Figure 9a shows that we counted the probability of nodes of Class 1 being elected as LEADER nodes in twenty rounds of election for four difficulty bits in HPoC. Among them, the node of Class 1 had the highest chance of being elected as LEADER node, around 0.7, and it did not change with the total number of nodes. This was in line with the design idea of this paper, i.e., that the more capable nodes are more likely to become LEADER nodes and take the responsibility of collecting Tx data and generating blocks. Finally, in Figure 9b, we simulated the transaction data generated by IoT devices with Jmeter, sent them to the HPoC consensus-based blockchain, and counted the transaction volume in the minute block. It was found that the data volume in the first minute was around 40,000, and the data in every minute after that were between 110,000 and 120,000. This is because the minute block only counts the data in one minute, and when the blockchain is just started, it will be in some middle time of one minute and the data sent before the start will be lost, causing the small transaction volume in the first minute. When the blockchain is successfully started and runs normally, its transaction volume per minute is above 110,000, which means its TPS is around 2000 and can meet the transaction volume demand of the IoT edge scenario.

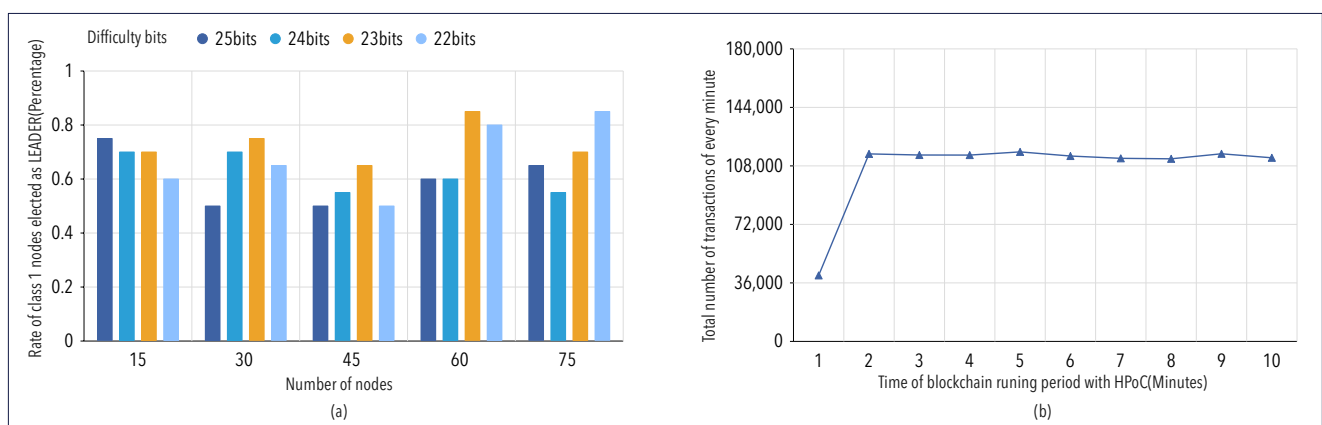


Figure 9. Simulation results: (a) probability of Class 1 nodes elected as LEADER; (b) tendency of transactions of running blockchain with HPoC in ten minutes.

Overall, the experiments designed in this section demonstrate that HPoC is a consensus in which the time consumption can be flexibly controlled; it possesses scalability and strong stability and is able to identify the capabilities of nodes. The blockchain system that was designed based on this consensus is able to reach 2000 TPS in transaction volume, which shows that HPoC can meet the demand of deploying a lightweight blockchain on heterogeneous devices in resource-constrained IoT edge-computing scenarios and proves the feasibility of the design idea in this paper.

7. Conclusions and Future Work

This paper addressed the mismatch between existing blockchain technology and IoT resource-constrained edge-computing devices and proposed a lightweight PoC consensus that can unify the storage, computing, and bandwidth resources of IoT edge devices for consideration and select more capable nodes responsible for efficient blockchain generation without high energy-consuming mining computation and consumption of financial-oriented tokens. Furthermore, a hierarchical structure and asynchronous proof-of-work mechanism, categorized by minutes, ten minutes, and days, were proposed to reduce node storage strain and increase the tamper-proofness of blockchain data. The comparative experiments between HPoC and PoW in this paper demonstrated the scalability, stability, and

ease of controlling the time consumption of HPoC. The statistics of blockchain transactions per minute prove that its TPS can meet the needs of existing IoT scenarios.

This study was based on Raft-type consensus, which is targeted to the crash fault tolerance (CFT) problem; nodes in the network will only experience downtime rather than evil conduct. HPoC's consensus-building mechanism will need to be modified in the future to improve its resilience to Byzantine fault tolerance (BFT) difficulties.

Author Contributions: Conceptualization, Z.N. and M.Z.; methodology, Z.N.; software, Z.N.; validation, Z.N., M.Z. and Y.L.; formal analysis, Z.N.; investigation, Z.N.; resources, Z.N.; data curation, Z.N.; writing—original draft preparation, Z.N.; writing—review and editing, Z.N.; visualization, Z.N.; supervision, Y.L.; project administration, Y.L.; funding acquisition, M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Key Research and Development Program of China (2019YFB2102403).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare that they have no conflicts of interest regarding this work.

References

1. Motherboard. How 1.5 Million Connected Cameras Were Hijacked to Make an Unprecedented Botnet. 2016. Available online: https://motherboard.vice.com/en_us/article/8q8dab/15-million-connected-cameras-ddos-botnet-brian-krebs (accessed on 23 May 2021).
2. Bursztein, E.; Cochran, G.J.; Durumeric, C.Z.; Halderman, J.A. Understanding the mirai botnet. In Proceedings of the 26th USENIX Security Symposium, Vancouver, BC, Canada, 16–18 August 2017.
3. He, W.; Golla, M.; Padhi, R.; Ofek, J.; Drmuth, M.; Fernandes, E.; Ur, B. Rethinking access control and authentication for the home internet of things. In Proceedings of the 27th USENIX Security Symposium (USENIX Security 18), Baltimore, MD, USA, 15–17 August 2018; pp. 255–272.
4. Fernandes, E.; Rahmati, A.; Jung, J.; Prakash, A. Decentralized action integrity for trigger-action iot platforms. In Proceedings of the 2018 Network and Distributed System Security Symposium, San Diego, CA, USA, 13–18 February 2018.
5. Zhou, W.; Jia, Y.; Yao, Y.; Zhu, L.; Guan, L.; Mao, Y.; Liu, P.; Zhang, Y. Discovering and understanding the security hazards in the interactions between {IoT} devices, mobile apps, and clouds on smart home platforms. In Proceedings of the 28th USENIX Security Symposium (USENIX Security 19), Santa Clara, CA, USA, 14–16 August 2019; pp. 1133–1150.
6. Chen, J.; Zuo, C.; Diao, W.; Dong, S.; Zhao, Q.; Sun, M.; Lin, Z.; Zhang, Y.; Zhang, K. Your iots are (not) mine: On the remote binding between iot devices and users. In Proceedings of the 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Portland, OR, USA, 24–27 June 2019; pp. 222–233.
7. Almakhdhub, N.S.; Clements, A.A.; Bagchi, S.; Payer, M. μ Rai: Securing embedded systems with return address integrity. In Proceedings of the Network and Distributed Systems Security (NDSS) Symposium, San Diego, CA, USA, 23–26 February 2020.
8. Zhou, J.; Du, Y.; Shen, Z.; Ma, L.; Criswell, J.; Walls, R.J. Silhouette: Efficient protected shadow stacks for embedded systems. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), Virtual, 12–14 August 2020; pp. 1219–1236.
9. Nakamoto, S.; Bitcoin, A. A Peer-to-Peer Electronic Cash System. 2008, Volume 4. Available online: <https://bitcoin.org/bitcoin.pdf> (accessed on 23 May 2021).
10. Puthal, D.; Malik, N.; Mohanty, S.P.; Kougianos, E.; Das, G. Everything you wanted to know about the blockchain: Its promise, components, processes, and problems. *IEEE Consum. Electron. Mag.* **2018**, *7*, 6–14. [[CrossRef](#)]
11. Mukherjee, M.; Matam, R.; Shu, L.; Maglaras, L.; Ferrag, M.A.; Choudhury, N.; Kumar, V. Security and privacy in fog computing: Challenges. *IEEE Access* **2017**, *5*, 19293–19304. [[CrossRef](#)]
12. Xie, X.; Pan, X.; Zhang, W.; An, J. A context hierarchical integrated network for medical image segmentation. *Comput. Electr. Eng.* **2022**, *101*, 108029. [[CrossRef](#)]
13. Tschorsch, F.; Scheuermann, B. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2084–2123. [[CrossRef](#)]
14. Yuan, Y.; Wang, F.-Y. Blockchain: The state of the art and future trends. *Acta Autom. Sin.* **2016**, *42*, 481–494.
15. Gervais, A.; Karame, G.O.; Wüst, K.; Glykantzis, V.; Ritzdorf, H.; Capkun, S. On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 3–16.

16. Xie, X.; Pan, X.; Shao, F.; Zhang, W.; An, J. MCI-Net: Multi-scale context integrated network for liver CT image segmentation. *Comput. Electr. Eng.* **2022**, *101*, 108085. [[CrossRef](#)]
17. Scaling byzantine agreements for cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai China, 28 October 2017; pp. 51–68.
18. Ongaro, D.; Ousterhout, J. In search of an understandable consensus algorithm. In Proceedings of the 2014 USENIX Annual Technical Conference (Usenix ATC 14), Berkeley, CA, USA, 19–20 June 2014; pp. 305–319.
19. Ali, M.S.; Dolui, K.; Antonelli, F. Iot data privacy via blockchains and ipfs. In Proceedings of the Seventh International Conference on the Internet of Things, Linz, Austria, 22–25 October 2017; pp. 1–7.
20. Liang, X.; Zhao, J.; Shetty, S.; Li, D. Towards data assurance and resilience in iot using blockchain. In Proceedings of the MILCOM 2017—2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 23–25 October 2017; pp. 261–266.
21. Gord, M. Smart contracts described by nick szabo 20 years ago now becoming reality. *Bitcoin Magazine*, 26 April 2016.
22. Conoscenti, M.; Vetro, A.; de Martin, J.C. Blockchain for the internet of things: A systematic literature review. In Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, Morocco, 29 November–2 December 2016; pp. 1–6.
23. Banerjee, M.; Lee, J.; Choo, K.-K.R. A blockchain future for internet of things security: A position paper. *Digit. Commun. Netw.* **2018**, *4*, 149–160. [[CrossRef](#)]
24. Saputro, M.Y.A.; Sari, R.F. Securing IoT network using lightweight multi-fog (LMF) blockchain model. In Proceedings of the 2019 6th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Bandung, Indonesia, 18–20 September 2019; pp. 183–188.
25. Srivastava, G.; Crichigno, J.; Dhar, S. A light and secure healthcare blockchain for iot medical devices. In Proceedings of the 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), Edmonton, AB, Canada, 11 October 2019; pp. 1–5.
26. Zhou, J.; Zhang, D.; Ren, W.; Zhang, W. Auto Color Correction of Underwater Images Utilizing Depth Information. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 1–5. [[CrossRef](#)]
27. Akkoyunlu, E.A.; Ekanadham, K.; Huber, R.V. Some constraints and tradeoffs in the design of network communications. In Proceedings of the Fifth ACM Symposium on Operating Systems Principles, New York, NY, USA, 19–21 November 1975; pp. 67–74.
28. Lamport, L.; Shostak, R.; Pease, M. The byzantine generals problem. In *Concurrency: The Works of Leslie Lamport*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 203–226.
29. Lamport, L. The part-time parliament. In *Concurrency: The Works of Leslie Lamport*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 277–317.
30. Castro, M.; Liskov, B. Practical byzantine fault tolerance. *OSDI* **1999**, *99*, 173–186.
31. Yang, F.; Zhou, W.; Wu, Q.; Long, R.; Xiong, N.N.; Zhou, M. Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism. *IEEE Access* **2019**, *7*, 118541–118555. [[CrossRef](#)]
32. Shoker, A. Sustainable blockchain through proof of exercise. In Proceedings of the 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 30 October–1 November 2017; pp. 1–9.
33. Xie, X.; Zhang, W.; Wang, H. Dynamic adaptive residual network for liver CT image segmentation. *Comput. Electr. Eng.* **2021**, *91*, 107024. [[CrossRef](#)]
34. Eyal, I.; Gencer, A.E.; Sirer, E.G.; van Renesse, R. Bitcoin-NG: A scalable blockchain protocol. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16), Berkeley, CA, USA, 16–18 March 2016; pp. 45–59.
35. Zhou, H.; Wu, T. Towards high accuracy pedestrian detection on edge gpus. *Sensors* **2022**, *22*, 5980. [[PubMed](#)]
36. Buterin, V.; Reijnders, D.; Leonardos, S.; Piliouras, G. Incentives in Ethereum’s hybrid Casper protocol. *Int. J. Netw. Manag.* **2020**, *30*, e2098. [[CrossRef](#)]
37. Micali, S.; Rabin, M.; Vadhan, S. Verifiable random functions. In Proceedings of the 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039), New York, NY, USA, 17–19 October 1999; pp. 120–130.
38. Hanke, T.; Movahedi, M.; Williams, D. Dfinity technology overview series, consensus system. *arXiv* **2018**, arXiv:1805.04548.
39. David, B.; Gazi, P.; Kiayias, A.; Russell, A. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, 29 April–3 May 2018; pp. 66–98.