DEAN: A Lightweight and Resource-efficient Blockchain Protocol for Reliable Edge Computing

***Abstract*—Edge computing draws a lot of recent research interests because of the performance improvement by offloading many workloads from the remote data center to nearby edge nodes. Nonetheless, one open challenge of this emerging paradigm lies in the potential security issues on edge nodes and end devices, e.g., sensors and controllers. This paper proposes a cooperative protocol, namely DEAN, equipped with a unique resource-efficient quorum building mechanism to adopt blockchain seamlessly in edge computing infrastructure to prevent data manipulation and allow fair data sharing with quick recovery under resource constraints of limited storage, computing, and network capacity. Specifically, DEAN leverages a parallel mechanism equipped with three independent core components, effectively achieving low resource consumption while allowing secured parallel block processing on edge nodes. We have implemented a system prototype based on DEAN and experimentally verified its effectiveness with a comparison with four popular blockchain implementations: Ethereum, Parity, IOTA, and Hyperledger Fabric. Experimental results show that the system prototype exhibits high resilience to arbitrary failures. Performance-wise, DEAN-based blockchain implementation outperforms the state-of-the-art blockchain systems with up to $88.6\times$ higher throughput and $26\times$ lower latency.**

***Index Terms*—Edge computing, Blockchains, Distributed computing, Consensus protocols.**

I. Introduction

A. Motivation

Edge computing [1] offers an efficient means to process collected data (e.g., through sensors and other end devices) at nearby edge nodes as opposed to transferring data back to remote data centers—a common practice in cloud computing. Edge computing saves the network traffic and improves the application performance, particularly for those latency-sensitive applications such as virtual reality [2]. Nevertheless, edge computing poses new technical challenges including security and privacy concerns with edge nodes and sensors [3]. The root cause of these concerns lies in the fact that existing security techniques used in data centers and cloud computing [4]–[6] are hardly applicable to the edge nodes and sensors. We highlight two outstanding discrepancies between cloud computing and edge computing in the following.

- **System Infrastructure.** The edge nodes and end devices constitute a loosely-coupled distributed system of highly heterogeneous participants with wireless connections. For instance, edge computing nodes range from smart phones to workstations mostly connected through wireless networking such as WiFi and ZigBee, whilst cloud computing data centers comprise racks of on-premises homogeneous blade-servers interconnected with high-speed network infrastructures.
- **Resource Constraints.** The edge nodes and end devices such as sensors are equipped with many constrained resources in terms of CPU, memory, storage, network, and power. Therefore, many assumptions well-accepted in data centers do not hold in edge computing. As a case in point, a typical sensor only has about 10-100 MB memory comparing with tens of gigabytes of memory available on the blade-servers in data centers. Other resource constraints in edge computing include but not limited to: no or small storage capacity, battery power, to name a few.

Due to the openness and resource-constraint security mechanisms of edge computing systems, various security incidents were repeatedly reported. One notable incident was that a Jeep SUV was remotely hijacked through its UConnect's cellular connection [7]. In addition, mobile devices are vulnerable to malicious applications to a large degree [8], e.g., the users install applications from untrusty third-party sources [9].

One plausible approach to tackle the security challenge in edge computing is to encrypt the edge data with stronger mechanisms such as public key infrastructure (PKI) [10]. Unfortunately, the computing capability of edge sensors is limited; even if they can carry out the encryption (of every single message), the power consumption is prohibitive as they are mostly battery-powered [11]. While a single node can hardly protect the data, the following approaches are based on the cooperation between multiple nodes. In *dependable systems*, there are a vast of studies derived from Paxos [12] and Practical Byzantine Fault Tolerance (PBFT) [13].

Both Paxos and PBFT require a lot of message passing, likely saturating the bandwidth of edge node's wireless network fairly quick. One notable variant is based on full replication: each node stores a replica of the data initiated by the leader and the integrity is guaranteed as long as no more than 50% of nodes are compromised. This approach requires less communication and instead takes more storage space. Of note, the latter approach reflects the design philosophy of public blockchains (Bitcoin [14] being, arguably, the most popular blockchain application). Nonetheless, an existing blockchain cannot be directly applied to an edge network: blockchains (or shards thereof) require all participants (full nodes) to store the entire history of the data provenance, which clearly cannot be accommodated by most end devices. *To summarize, existing security and fault tolerance approaches pose unrealistic expectations on compute, network, or storage in the context of edge computing.*

If we reexamine the aforementioned approaches to trustworthy edge computing, the blockchain-based approach has the potential to become a viable solution: storage or space constraints might be overcome by leveraging software approaches such as compression and shallow replication. Therefore, various fields have spent much effort in investigating

TABLE I
FEATURE COMPARISON AMONG EXISTING BLOCKCHAIN PROTOCOLS AND THE PROPOSED PROTOCOL, DEAN.

Features	Hyperledger [15]	Ethereum [16]	Parity [17]	Omniledger [18]	RapidChain [19]	IOTA [20]	DEAN
IoT **specific** blockchain	×	×	×	×	×	✓	✓
Lightweight consensus protocol	×	×	×	×	×	×	✓
IoT-specific quorum selection policy	×	×	×	×	×	×	✓
IoT-specific reliable sharded chain	×	×	×	×	×	×	✓
Resource-efficient data replication	×	×	×	✓	✓	✓	✓

taking blockchain or its variants for their application-specific needs. For instance, "with the advent of blockchain, decentral[ized] data management can be implemented in a privacy-preserving and efficient way," recently stated car manufacturer BMW [21]. However, blockchain at the time of writing this paper is mostly used as a blackbox without a clear pathway to overcome the aforementioned obstacles: Although blockchain has proven its high security in cryptocurrency, the consensus protocols taken by all popular blockchains assume abundant resources (in terms of computation, storage, and network) of the system infrastructure. New consensus protocols must be designed and tailored to the specific characteristics of edge nodes and end devices. Table I summarizes the most outstanding limitations (more discussion in Section II) from the existing blockchain systems in edge computing and the Internet-of-Things (IoT) ecosystem.

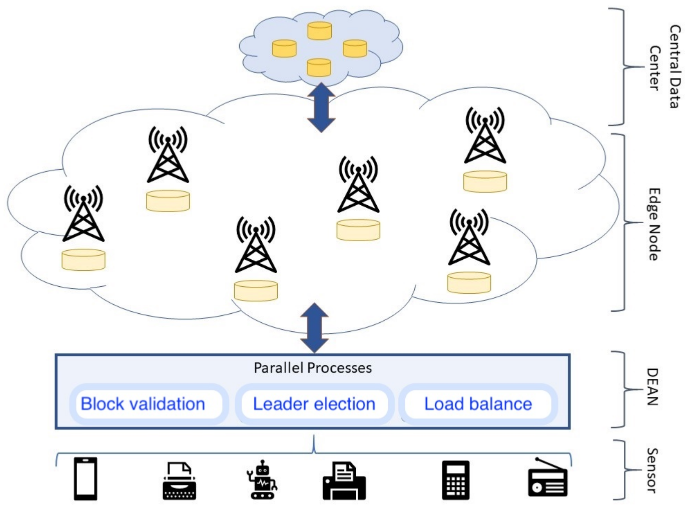


Fig. 1. Four-tier edge computing architecture with DEAN.

B. Proposed Solution

This paper proposes a set of new blockchain protocols for edge nodes (along with end devices) under resource constraints. We name the edge network powered by the new protocol DEAN: Decentralized-Edge Autonomous Network, as illustrated in Figure 1. The key idea of our decentralized protocols is threefold: (i) leveraging the resources (i.e., persistent space and computational power) available in the edge nodes to ameliorate the pressure on the end devices (e.g., edge sensors); (ii) designing a lightweight but reliable consensus protocol to support scalability and fast processing of client requests; and (iii) balancing the storage pressure among edge nodes that enables fair sharing of the data. In addition to the above requirements, the new protocol must ensure, with provable guarantees, that the entire system's data integrity is not tampered with.

One concrete example of applications that can benefit from the proposed blockchain-based edge computing platform is supply chains [22]. The security threat in supply chains usually arises from counterfeiting, which is a serious concern of the government and the industry [23]–[26]. The conventional centralized approach for supply chain management exhibits questionable reliability: for example, a successful deny-of-service (DoS) attack [27] on the single node will suspend the entire service. From the above discussion, it is evident that a decentralized system such as a lightweight blockchain can potentially provide more reliable product verification and transaction management for supply chains.

In addition to exploiting lightweight consensus protocols, we will demonstrate that the proposed design achieves high efficiency: Experimental results show that the system prototype built upon DEAN outperforms mainstream blockchain systems (Ethereum [16], Parity [17], IOTA [28], and Hyperledger Fabric [15]) by orders of magnitude in terms of both throughput and latency, which is attributed to the parallelism of block processing in DEAN.

In summary, this paper makes the following contributions:

- We characterize the edge computing ecosystem with a series of modules that are naturally aligned with blockchain design principles. (§III-A)
- We propose a lightweight consensus protocol to adopt blockchains in edge computing that supports IoT-specific energy-efficient quorum building and block processing in parallel, which allows us to scale the system out to thousands of nodes while keeping low the latency of processing client requests. Furthermore, we propose a parallel data dissemination strategy that not only maintains the equitable sharing of the storage among the edge nodes but also enables recovery of and fast accesses to the data even when in front of edge node failures. (§III-B)
- We carry out a thorough analysis of the proposed protocols in terms of both time complexity, spatial overhead, and number of messages. (§III-C)
- We implement the proposed protocol and optimization for edge computing and evaluate it with extensive experiments by including comparative study against state-of-the-art blockchain systems. (§IV)

II. Problem Statement and Challenges

The research question this paper thrives to answer is how to extend existing blockchain technology for data trustworthiness with limited resources in the unique infrastructure of edge computing. There are various challenges to be addressed. The system infrastructure in edge computing is a completely different story than that of workstations and clusters of servers: a large portion of the system is comprised of sensors that have limited computation power, negligible storage capacity, and wireless network connections. Though the edge nodes comparatively have more resources (i.e., computation power, disk space, and bandwidth), it is still challenging to deploy the traditional blockchain systems due to the unstable nature of wireless connections, extensive storage requirements, and energy consumption. To be more specific, the edge nodes can come and go arbitrarily, leading to inconsistency in blockchain operation anytime. Ideally, *the blockchain crafted for edge computing should be operated among the trustworthy nodes elected through an IoT-specific quorum selection policy so that the entire blockchain remains stable and incurs lightweight resource usage both on the edge nodes and sensors in terms of computation, storage, and networking*.

Table I illustrates that neither conventional public blockchains (high computation power), private blockchains (high network traffic) nor the sharded-based approach meet our goal. Present protocols (e.g., proof-of-work, practical byzantine fault tolerance, or hybrid) are highly resource-intensive either in computation or communication. The proof-of-stake appears to be suitable to apply in edge computing as it offers low complexity of communication and computational work. However, due to several reliability issues (e.g., nothing-at-stake, or Sybil attack due to the possibility of initial stake procurement from the public pool), it does not meet the requirements to apply directly to edge computing yet. Besides, as the edge nodes can arbitrarily come and go offline, in the sharded-based approaches the blockchain integrity might break at any time even with a few nodes failure. *Most importantly the current protocols are not equipped with the IoT-specific lightweight quorum selection mechanism that could assist in building a reliable blockchain network with the most trustworthy edge nodes*. To make it worse, all existing blockchain paradigms, no matter public, private, or a variant/hybrid thereof, require an unrealistic storage capacity on the nodes.

Although a recent work [29] focuses on edge computing at a small scale, it follows an expensive pre-computation to decide which edge devices will store the block before a traditional proof-of-stake supported mining process starts. Besides, the approach solely depends on the traditional proof-of-stake consensus mechanism that is vulnerable to several attacks (e.g., Sybil attack, nothing-at-stake). Moreover, a few more works [30], [31] design protocols that attempt to minimize the resource-intensiveness of either PBFT or PoW; yet, these still require a significant amount of energy both in terms of consensus and storage management.

To summarize, none of the current protocols offer an IoT-specific blockchain management technique that provides a precise balance among the reliability, computation, and storage requirements while affording scalability among the thousands of nodes. Therefore, the goal of the proposed protocols presented in this paper is to provide a low-complex and reliable consensus mechanism tailored with a space-efficient data replication strategy that is capable enough to scale up to the thousands of nodes without compromising the safety. To this end, we design new protocols to allow blockchains work with ***(i)*** among the most trustworthy nodes elected through an unique IoT-specific quorum election policy ***(ii)*** *limited* computation power and network bandwidth and ***(iii)*** *relaxed* storage requirements.

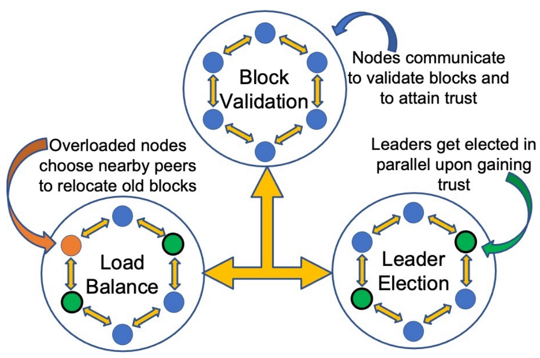

Block Validation
Nodes communicate to validate blocks and to attain trust
Overloaded nodes choose nearby peers to relocate old blocks
Leaders get elected in parallel upon gaining trust
Load Balance
Leader Election

Fig. 2. Three parallel processes work together in DEAN.

A. Threat Model

We consider the worst case in which an internal adversary has got authenticated to an edge node. With the authorized access to the edge devices, the attacker may attempt to alter or modify a transaction record, commit a false transaction (i.e., fraud), perform a denial-of-service attack on other users through an artificial escalation of security level, or even send unauthenticated messages. To be more specific, we are interested to see if the internal adversary can compromise data in 51% edge devices powered by the proposed mechanism.

Another crucial challenge in DEAN is forging of node attributes (e.g., adjacency list, degree of adjacency, location, token, or timestamp, etc.) to achieve unfair share. Nodes can closely monitor each other and check the attributes that are periodically updated. The attribute table is shared with the majority (i.e., at least 51%) of the adjacent nodes through the *Block-Validation* process (i.e., Protocol 1). Therefore, any inconsistent operation will help the fellow nodes to identify the malicious activities and eliminate the faulty nodes from the trusted node list.

III. Decentralized-Edge Autonomous Network

A. System Components

In this section, we will explain the three parallel core elements of DEAN protocol, as shown in Figure 2.

1) Block validation: Due to the typical nature of the consensus protocols (i.e., compute-intensive, communication-intensive, or Sybil attack due to initial token procurement), the traditional blockchain systems are not directly applicable in the resource-constrained edge computing environment. Thus, we build a protocol that holds both attributes: (i) reliability and (ii) lightweightness. The DEAN consensus mechanism

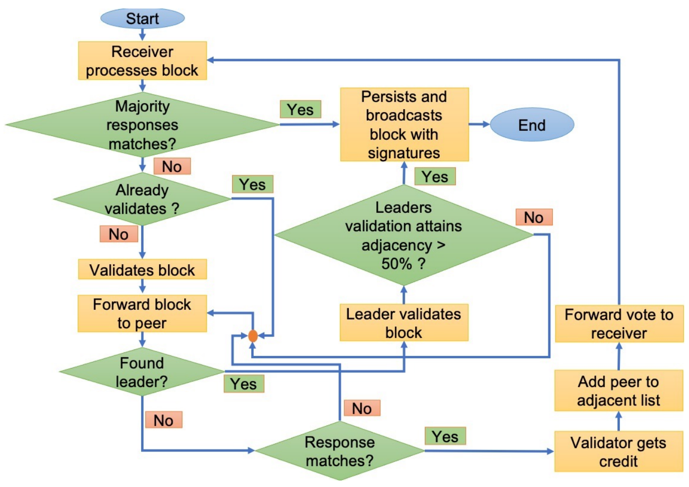


Fig. 3. Block validation continues while leaders are being elected in DEAN.

works through two parallel steps: (i) validates a block through peer nodes while developing the trust till the leaders get elected, (ii) builds the network of trustworthy leader nodes (see Section III-A2).

Figure 2 illustrates an overview, edge nodes participate in block validation with randomly chosen nodes while establishing the most trustworthy leaders in parallel. The nodes continue the block processing in parallel while developing the leaders through n number of phases (more discussion in Section III-A2). The initial participants are sorted based on the metadata such as degree of adjacency, token, timestamp, and location, etc. For this initial verification of the trustworthiness, nodes need to share their metadata information such as location, timestamp, and degree of adjacency periodically. Therefore, if the initial network size is large, and there is no available leader node nearby, a block receiver node picks a sub-cluster of $2f+1$ nodes based on the metadata as mentioned earlier to forward the block, where f is the maximum faulty nodes allowed in a sub-cluster. The receiver node continues to expand the initial cluster until the block attains consensus from more than 50% nodes in the entire network.

As shown in Figure 3, the receiver node validates the block and forwards the block to the peers for further validation. When a sufficient number of leaders are elected, the nodes stop exchanging excessive messages and continue to the next phase where the transaction validators list becomes minimal, such as randomly selected leader nodes can approve any new transaction. If the receiver finds enough leaders to validate, the block will not be broadcast among the general peers. If more than 50% peers provide valid votes or sufficient leaders provide consensus that collectively are adjacent to more than 50% trustworthy nodes, the block is then persisted and broadcast as a valid one. At the end of the validation process, the validators receive the credit, and both the validators and the first node (i.e., receiver) will add themselves as trusted peers.

Multiple leaders can work in parallel to continue validating separate sets of blocks. However, DEAN ensures that no leader attempts to mine the same block based on a locking mechanism on a block that ensures atomic operation. Each block has a creator hash (*tHash*) built with the timestamp assigned by a receiver node. Therefore, any node can verify who is the first or original miner of the block from the hash. The consensus mechanism will be discussed in more detail in Protocol 1.

2) Leaders election: To avoid the problem in the conventional protocols, DEAN is equipped with an IoT-specific quorum selection policy that leverages the scale-free network development mechanism to build a reliable blockchain network in an edge computing ecosystem with the most reliable nodes known as leaders. The proposed technique offers a double-edged solution: ***(i)*** a reliable and lightweight IoT-specific leader selection protocol; ***(ii)*** build an unique blockchain replication model to guarantee the cost-effectiveness and stability of the blockchain across the distributed nodes.

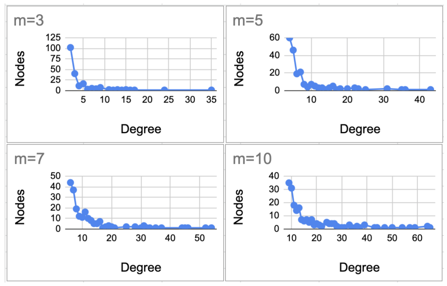
Scale-free network comes with several benefits, such as (i) reachability, (ii) reliability, and (iii) stability . In large scale-free networks the small world principle [32] may often hold. This principle says that any node is on average connected to any other node in a small number of steps. The nodes with many connections act as a kind of hub between all the other nodes. Large scale-free networks are not vulnerable to random attacks at nodes. They are vulnerable to targeted attacks at the few highly connected nodes. But the network will often not lose connections if some hubs fail, due to the fact that other hubs remain.

A network is called scale-free when the characteristics of the network remains independent with the scaling of nodes. That means that when the network grows, the underlying structure remains the same. The core property of a scale-free network **[33], [34]** is that it follows power-law degree distribution mechanism. In a scale-free network, new nodes tend to connect to existing ones with linear probability in the degree of the existing nodes. If this probability is sub-linear, all new nodes tend to connect with every other nodes and hubs are much smaller. If the probability is super-linear all new nodes will attempt to connect to few nodes with highest degree; hence, most of the nodes get centralized due to few hubs. Therefore, for simplicity, to avoid both scenarios, in a network of size n, we consider the probability with which a node connects or attempts to forward a block for validation to an existing one follows a power function of the existing nodes degree k:

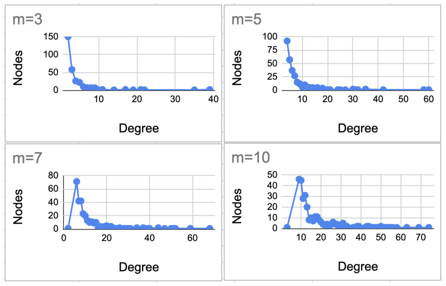
$$\pi(k) = A + k^{\alpha} \tag{1}$$

where A is the initial attractiveness of the node, $m >= \alpha > 0$, m is an arbitrary integer number, and $\log n < k < n$ to distribute the nodes across the network in uniform manner. The value for A is decided based on the node's metadata such as location, timestamp, degree, etc. Note that, if $\alpha < 1$ then it is sub-linear and if $\alpha > 1$ then it is super-linear. To be more specific, superlinearity leads to centralization, where very few nodes develop the leadership by acquiring the most adjacency. Therefore, we need to find an optimum value for m that will assist in building a reliable distributed network of leaders with a reasonable balance of adjacency among the leaders.

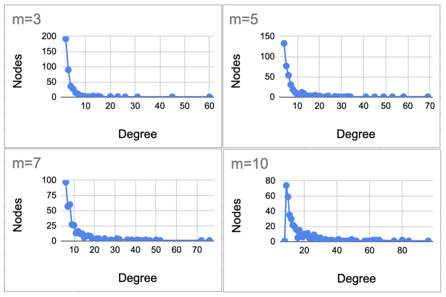
To find an optimum m, we implement the equation 1 in DEAN and observe the distribution of adjacency when a node tends to forward a block validation request to the trustworthy nodes at various scales for different values of



(a) 200 nodes



(b) 300 nodes



(c) 400 nodes

Fig. 4. Finding the optimal m to avoid sub-linear or super-linear nature of the nodes' adjacency. $7 < m <= 10$ produces more than 50% leaders with reasonable adjacency at various scales.

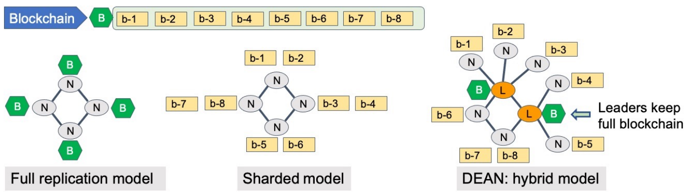


Fig. 5. Comparing various blockchain management **approaches** against DEAN.

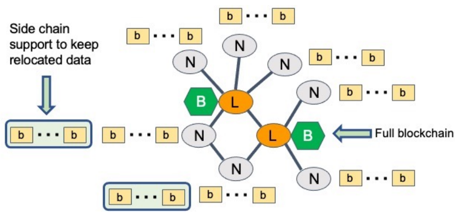


Fig. 6. Trustworthy nodes assists in load-balancing through side chain.

m. Figure 4 depicts that at different scales, the optimum value for m lies within a range $7 < m \leq 10$ that produces more than 50% nodes with leadership badge while avoiding excessive adjacency between every other node. Note that this leadership-building process is independent of the block validation process. However, finding a policy of an optimum m could be explored further and is part of another research scope.

The process of building the trusted leaders continues till at least 50% nodes achieve the leadership badge or a reasonable number of adjacencies based on Equation 1. A node gains a leadership badge if it meets the leader selection attributes such as *high degree*, *greater token*, *timestamp*, etc, and contains at least $2 \times (2f + 1)$ adjacency, where $(2f + 1)$ is an initial size of a sub-cluster of nodes. Once a few leaders are elected, the sorted leaders based on their attributes (e.g., *degree*, *token*, etc.) will be picked to serve in order based on their availability. *The rationale is a block will attain consensus quickly by exploring a few leaders when more than 50% leader nodes attain majority trusts* . To be more specific, exploring a leader is sufficient enough than exploring a group of nodes connected to that leader; hence, the consensus process remains much simpler and faster.

3) Load Balance: Figure 5 depicts, in a vanilla blockchain system, the blockchain is either fully replicated or sharded among all the nodes in a network. In some blockchain systems, the entire blockchain is sharded [35] among the peers in a straight-forward fashion. Due to the several limitations, the traditional mechanisms will be impractical to apply in edge computing: (i) the edge devices have limited storage, (ii) the edge nodes arbitrarily come and go. Therefore, *the blockchain data in edge devices should be organized in such a way that the blockchain remains available despite a bunch of nodes' unavailability and does not overwhelm the disk space*.

Figure 5 illustrates, through a hybrid model, DEAN allows storing the full blockchain only within the most trustworthy nodes known as leaders and manages the shards of a blockchain among the nearby peers of a specific leader. This smart sharding mechanism backed by the scale-free graph **development** approach offers a three-fold solution: *(i)* blockchain remain intact, even with the failures of a group of nodes, *(ii)* consensus process eventually turns faster as soon as a group of leaders gets elected, and finally *(iii)* due to storage limitation, if an edge device encounters disk overload, it further disseminates the oldest blocks to the sidechains of the fellow nodes to get more space in the disk backed by a quick data recovery mechanism, as shown in Figure 6. We will discuss the data dissemination technique in detail in Protocol 4.

B. Protocols

Our proposed consensus protocol, namely Decentralized-Edge Autonomous Network (DEAN) consists of four sub-protocols. The first protocol steers both the consensus and the distributed network construction through a scale-free graph **development** approach. The second protocol assists in finding leaders from the edge devices. The third protocol allows edge nodes to extend the blockchain network by approving new nodes to join. Finally, the fourth protocol assists with smart data distribution and recovery in case of edge devices failure (e.g., storage overload); hence, it helps in the successful continuation of the block mining process.

Block Validation. The Protocol 1 explains the entire consensus mechanism and process of building the distributed trustworthy network. In essence, when a node receives a batch of new transactions packed into a block propagates the block among the edge nodes. As shown in Line 5, the receiver node attempts to validate the block. After validation, the node starts looking for leader nodes to attain quickly more than 51% votes (Line 6) through their combined adjacency with the general nodes. In the absence of enough leaders, the block traverses the network to find available peers (Line 8). The validators

Protocol 1 `Block-Validation`

Require: N_i the total initial edge nodes in a network N; b the new block; b_n^i is the total adjacency of a block; b_l is the set of leaders for a block; b_{ln} is the total adjacency of a block through a set of leaders; Edge nodes E where the i-th node is E^i; E_B^i the blockchain copy on E^i; E_N^i the adjacent list on E^i; n the new node; f is total faulty nodes allowed in a sub-network with $2f+1$ nodes.

Ensure: b is validated by the majority of edge nodes.

```
1:  function DEAN-CONSENSUS($b^i$, $E$, $N_i$)
2:      $N_i \leftarrow 2f+1$
3:      for $E^i \in N_i$ do
4:          while $|b_n^i| \leq \frac{N}{2} || \; |b_{ln}^i| \leq \frac{N}{2}$ do
5:              if $E^i$ validates $b$ & $b \notin E_B^i$ then
6:                  $L \leftarrow$ FIND-LEADER $(b^i, E^i, N_i)$          ▷ Protocol 2
7:                  if $L$ is $\emptyset$ then
8:                      $L \leftarrow$ nearby peer $E^j$
9:                  else
10:                     $b_l^i \leftarrow b_l^i \cup L$
11:                 end if
12:                 $v \leftarrow$ BUILD-NETWORK $(b^i, L, N_i)$
13:                 if $v$ is valid & $E^j \notin E_N^i$ then
14:                     $E_N^i \leftarrow E_N^i \cup E^j$
15:                     $b_n^i \leftarrow b_n^i \cup E^j$
16:                 end if
17:                 if $|b_n^i| \geq \frac{N}{2} || \; |b_{ln}^i| \geq \frac{N}{2}$ then
18:                     $E_B^i \leftarrow E_B^i \cup b$          ▷ Commit block
19:                     Forward final result to peers
20:                 end if
21:             end if
22:         end while
23:     end for
24: end function
25: function BUILD-NETWORK($b^i$, $E^j$, $N_i$)
26:     if $b^i \notin E_B^j$ & $Verification\ Matches$ then
27:         if $E^i \notin E_N^j$ then
28:             $E_N^j \leftarrow E_N^j \cup E^i$
29:             $E_{tok}^j \leftarrow E_{tok}^j + 1$
30:         end if
31:     end if
32:     if $b_n^i \geq \frac{N}{2} || \; |b_l^i| \geq \frac{N}{2}$ & $b \notin E_B^j$ then
33:         $E_B^j \leftarrow E_B^j \cup b$          ▷ Commit block
34:     end if
35:     Forward vote $v$ to sender
36: end function
```

(e.g., leaders or peers) attempt to validate the block (Line 12). If the block is valid, the receiver node and the validators add each other as trustworthy peers and share the transaction fee as shown in Line 13 and Line 26. The block gets persisted if the votes reach the minimum limit (i.e., 51% adjacency) either through the general peers or combined adjacency score from a set of leaders (Line 18 and Line 33).

The consensus mechanism (i.e., block validation) is independent of the *Leader Election* process. To be more specific, a block validation continues at any time even though in the absence of leaders; however, the validation process gets faster when sufficient leaders become available through the *build-network* mechanism. In our blockchain system, each node will have an attribute table that is shared and updated among the peers periodically. The table holds the most recent status of the different attributes of a node such as degree of adjacency, token, timestamp, and location, etc. Therefore, if a node becomes compromised at a certain point, the majority of the fellow nodes already have the updated attributes that help in verifying the messages sent from the compromised node. However, if a node denies sharing its table status with other nodes within a fixed clock period, the node is broadcast as a faulty one, and no further communication is allowed from it.

Protocol 2 `Find-Leader`

Require: N_i the total initial edge nodes; b the new block; b_{tHash} the unique identifier of first miner of the block; b_l is the set of leaders for a block; Edge nodes E where the i-th node is E^i; Leader nodes L where the i-th node is E_l^i; L is a leader node.

Ensure: Most trustworthy node L gets elected.

```
1:  function FIND-LEADER($b$, $E^i$, $N_i$)
2:      while new block arrives do
3:          if $b_l^i$ is not $\emptyset$ then
4:              Move to a disjoint set of nodes
5:          end if
6:          $E^j \leftarrow$ Find nodes with $k^\alpha$ degree
7:          if $E^j$ is $\emptyset$ then
8:              $E^j \leftarrow$ Find nodes with max token
9:          else
10:             $E^j \leftarrow$ Find nearby peer with max adjacency
11:         end if
12:         $L \leftarrow E^j$
13:         if $L \notin b_l^i$ then
14:             Forward $L$
15:         end if
16:     end while
17: end function
```

Find Leader. Protocol 2 assists in finding leaders or the most trustworthy nodes. The other protocols take the advantage of the Protocol 2 when in need of finding leader nodes. As the leading building process is parallel and develops during the block validation process, we do not expect to have leaders at the very first stage. However, once there are a sufficient number of leaders get developed, both the block validation and the blockchain persisting processes become more efficient.

At first, the protocol attempts to find a leader based on equation 1, as shown in Line 6. If no leader is located, which is the case at the very initial stage, the protocol then looks for the other criteria, such as max token (Line 8) or max adjacency (Line 10) to find out the most reliable node. The protocol keeps track of whether the same leader is being elected every time (Line 13). It (i.e., Protocol 2) prohibits the leader election from the same set of nodes and prefers to choose a leader each time from a disjoint set of nodes (Line 4). Choosing leaders from a disjoint set of nodes each time keeps the entire validation process undoubtedly reliable.

Protocol 3 `New-Node-Approval`

Require: N_i the total initial edge nodes; b the new block; Edge nodes E where the i-th node is E^i; n the new node; n_s the trustworthiness flag of new node; n_h the hash of new node; n_l is the set of leaders for the new node; L is a leader node.

Ensure: n is verified before it is added to network.

```
1: function JOIN-NEWNODE($b$, $E$, $N_i$, $n$)
2:     while $|n_l| <= 2$ do
3:         $L \leftarrow$ FIND-LEADER$(b, n, N_i)$          ▷ Protocol 2
4:         if $n$ validates $b$ then
5:             compute $n_h$
6:         end if
7:         if $L$ verifies $n_h$ & $L \notin n_l$ then
8:             $L \leftarrow L \cup n$
9:             $n_l \leftarrow n_l \cup L$
10:            $L$ earns joining fee from $n$
11:        end if
12:        if $|n_l| >= 2$ then
13:            $E$ accepts $n$ as new node
14:        end if
15:    end while
16: end function
```

New Node Approval. Avoiding the unnecessary communication overhead during the *Build-Network*, along with the growth of the network, is a critical issue. In DEAN, we propose a mechanism to address this point. Once the leaders are elected, the new nodes do not need to communicate extensively with all the nodes to develop trust. Instead, they can join the blockchain network based on the vote from a set of leaders as shown in Protocol 3. That is, new nodes will first try to achieve trust from the leaders.

As shown in Line 3, the new node will try first to find a qualified leader before joining the network. The leader forwards a block to the new node to verify (Line 4). The new node then shares the result with the leader after validating the block. The leader approves the node in the network if the result is valid (Line 7) and earns a joining fee (Line 10). The process continues as long as the new node attains trust at least from two disjoint leaders (Line 12). When other edge devices find votes from the leaders, they automatically add the new node into their trusted adjacent list (Line 13).

Load Balance. We explain the process of data dissemination and recovery in case of disk overload in a node in Protocol 4. As shown in Line 2, an edge node can start disseminating the oldest blocks to nearby fellow nodes if the disk space is occupied by more than 80%. Note that at the early stage (i.e., Protocol 1), after validating a block, each leader stores the hash pointers (i.e., *rList*) of adjacent nodes before disseminating the block to the fellow nodes. Therefore, to disseminate the old blocks, the edge nodes select only those fellow nodes that are not available in the *rList* (Line 7). The nearby disjoint leaders or the most trustworthy nodes get the highest priority for a relocated block (Line 4).

The sender node sets the block relocation flag before disseminating (Line 6) to distinguish between a regular block and an old block. As shown in Line 25, first, the receiver node will double-check if the block is sent for relocation purposes and is not already stored in the local blockchain. Before storing the block in the disk, the receiver node will first verify the source node's hash (Line 25). If the hash is valid, the node will compute a new hash (i.e., t_b^h) (Line 26) for the relocated block and store it (i.e., block) along with the current hash (i.e., *cHash*).

We use a secured side chain in each node to store the relocated blocks to keep them separate from the regular (i.e., already mined) blocks. The blocks in a side chain can always be verified from the *rList* stored in the source node. The node will store the relocated block in a side chain (Line 27) before responding to the sender node (Line 28) with the new relocated hash (i.e., t_b^h). The sender node will include the new hash in

Protocol 4 `Load-Balance`

Require: Edge nodes E where the i-th node is E^i; Adjacent nodes list E_a where the i-th node is E_a^i; E_B^i the blockchain copy on E^i; E_{disk}^i the disk space on E^i; E_{BT}^i the side blockchain of E^i; E_{bp}^i the block pointer to adjacent edge node; b the oldest block; b_h the hash of a block; b_t the flag to decide block location; t_h^b the hash pointer of relocated adjacent edge node.

Ensure: Overloaded node E_i shares block with reliable nodes.

```
1: function DISSEMINATION($b$, $E$)
2:     if $E_{disk}^i >= 80\%$ then
3:         while $c_l <= 2$ do
4:             $L \leftarrow$ FIND-LEADER $(b, E^i, N_i)$ ▷ Protocol 2
5:             if $L$ is not $\emptyset$ & $L_{disk} <= 80\%$ then
6:                 $b_t \leftarrow True$          ▷ Block for side chain
7:                 if $L \notin E_a$ then    ▷ Not in adjacent nodes
8:                     STORE-BLOCK$(b, E^i, L)$
9:                     $c_l \leftarrow c_l + 1$
10:                end if
11:            end if
12:        end while
13:    end if
14: end function
15: function STORE-BLOCK($b$,$E^i$,$E^j$)
16:    $t_h^b \leftarrow$ RECEIVE-BLOCK$(b, E^j)$
17:    if $t_h^b$ is valid then
18:        $E_{bp}^i \leftarrow E_{bp}^i \cup t_h^b$              ▷ Keep pointer
19:        Empty the data from block $b$
20:        Split rewards with $E^j$
21:    end if
22: end function
23: function RECEIVE-BLOCK($b$, $E^j$)
24:    while Block $b$ arrives do
25:        if $b_t$ & $b \notin E_{BT}^j$ & $b_h$ confirms source then
26:            Compute $t_h^b$
27:            $E_{BT}^j \leftarrow E_{BT}^j \cup b$          ▷ Add to side chain
28:            Return $t_h^b$
29:        end if
30:    end while
31: end function
```

TABLE II
COMPLEXITIES OF VARIOUS BLOCKCHAIN SHARDING PROTOCOLS.

Protocols	Time	Space
RapidChain	$O(n^2)$	$O(m \cdot \lvert B\rvert/n)$
Omniledger	$O(n^2)$	$O(m \cdot \lvert B\rvert/n)$
DEAN	$O(\log n)$	$O(L \cdot \lvert B\rvert + (n-L) \cdot \log \lvert B\rvert)$

the block's relocation hash pointer list (i.e., *rList*), as shown in Line 18. Finally, the sender node prepares a hollow block case for the transferred block. That is, it (i.e., sender) keeps only all the hash values of the block for future quick data recovery purposes and erases the data content from the disk to make more space (Line 19). As the receiver node helps in relocating the block, it splits the reward for the block with the sender node, as shown in Line 20. The entire process continues as long as at least two reliable nodes replicate the block.

C. Analysis

1) Security Guarantee: **The security of conventional blockchains relies on the one-way function presumably realized by (cryptographical) hash functions such as SHA-256. DEAN does not introduce new hash functions, and therefore the security guarantee of DEAN-based blockchain implementations is on par with that of mainstream blockchains as long as we can *simulate* DEAN with well-known consensus protocols such as PoW or PBFT. While a full mathematical proof of simulating DEAN with PoW is beyond the scope of this paper, in the following we sketch the construction of such a simulation by showing that the extra information exposed by DEAN is negligible[1] according to complexity theory [36]. We by n denote the number of nodes in the blockchain network. PoW incurs $O(n)$ messages from the 1-to-n block broadcasting. Let m denote the number of leading nodes, then the voting procedure among m leading nodes incurs $O(m)$ messages. If the voting is propagated outwards all the n nodes, then the number of messages in this phase is $O(mn)$. The overall number of messages is $O(m) + O(mn) = O(mn)$. That is, DEAN introduces up to $O(m)$-fold more messages than PoW, meaning that the adversary might observe up to $O(m)$ more $\langle$plaintext, ciphertext$\rangle$ pairs. Let N denote the message length (in bit-string), i.e., the block size in the blockchain. It is easy to see that the additional $O(m)$ messages are *negligible* comparing with the entire message space because $m \ll 2^N$. To get a quick idea of m and 2^N, in Bitcoin the block size is 1 MB meaning that $2^N = 8,000,000$, $N = \log 8,000,000 \approx 300$, while m is a much smaller number, e.g., $m \leq 10$, cf. Fig. 6.**

2) Time complexity: Without loss of generality, we calculate the complexity of DEAN and compare against the vanilla sharded-based approaches to analyze the resource efficiency by DEAN, as shown in Table II. For validation of a block, DEAN initially starts with $m = 2f + 1$ number of nodes and eventually expands the cluster to attain more than 50% votes among a total network of size n, where f is an arbitrary number and denotes the maximum faulty nodes allowed in a $2f + 1$ cluster. Eventually, in parallel, DEAN dynamically develops some leaders L, connected to a set of random nodes from the edge network k that jointly present more than 50% trustworthy adjacency.

Let tx denote a transaction that belongs to a dynamically constructed cluster m. First, the sensor sends tx to the edge receiver node. The receiver validates and routes the tx to an available cluster m, which incurs a computation overhead of $O(m)$, where $m \approx \log n$. The nodes in the responsible cluster validate the transaction before communicating with the client with the validation result (i.e., encrypted vote) with m messages. Finally, the client aggregates the votes and responds to the DEAN server at a constant rate. Therefore, the communication and computation overhead for a transaction in total is $O(m + m + 1) = O(m) = O(\log n)$.

3) Space complexity: Let $|B|$ denote the size of the blockchain. In DEAN, not every node replicates the entire blockchain. Rather, only the leader nodes store the full blockchain, whereas, the other nodes will only hold a fraction of the blockchain, which relaxes the disk requirements to a large extent compared to the proof-of-work (PoW) and practical byzantine fault tolerance (PBFT) protocols. Besides, this hybrid model brings more reliability compared the vanilla sharding mechanisms, because, even with a failure of a fractions of nodes, the entire blockchain network remain intact. Therefore, the space requirement for storing the blockchain powered by DEAN is only $O(L \cdot |B| + (n - L) \cdot \log |B|)$. Whenever a node receives a new block, the adjacent leaders synchronizes the local ledgers. As the nodes do not perform peer-to-peer communication during the synchronization phase, the cost of data synchronization lies within a reasonable limit as it requires only $O(\log n)$ messages.

IV. EVALUATION

A. Experimental Setup

We have implemented a prototype system of the proposed DEAN blockchain architecture and consensus protocols with Python and deployed to a cluster comprised of 58 nodes interconnected with 802.11n communication protocol. Each node is equipped with an Intel Core-i7 2.6 GHz 32-core CPU along with 296 GB 2400 MHz DDR4 memory; hence, each node can be emulated with up to 32 nodes. We emulate multiple edge nodes with Docker [37] and distribute them equally among the physical nodes. A docker container represents either as an edge node or a sensor node, and nodes communicate with each other through a standard socket interface. Sensor nodes mainly forward transaction from client to edge nodes for further processing. The ratio between edge and sensor nodes is set to 1:3 by default, i.e., $\frac{|S|}{|M|} = 3$. The main workload under evaluation comprises 2.8 million queries similar to *YCSB* [38]. *YCSB* is widely accepted in measuring the performance of blockchain systems (e.g., BlockBench [39]).

B. Fault Tolerance

In this section, we experimentally verify the **fault tolerance** of the proposed protocol. The approach we take is to feed the

[1]There is a more formal definition of *negligible function* in cryptography using polynomials.

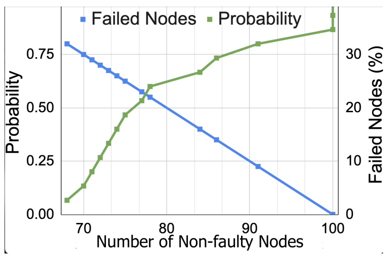


Fig. 7. **DEAN's Fault Tolerance.** DEAN guarantees that more than 50% of nodes remain intact and hold valid blockchain in practice in front of various number of nodes failure.

system with a large number of random transactions extracted from YCSB [38] and let the system run for 180 minutes in a 1000-node cluster. We repeated the experiments 15 times. During each run, we randomly chose a number of edge nodes that either deny to provide validation service (i.e., denial of service attack) or inject false message during broadcasting the block validation result. To be more specific, we would like to observe that even with the failure of a number of nodes, the remaining network still holds the correct blockchain.

We report the results in Figure IV-A that shows all of the 15 executions lead to more than 68% nodes holding the correct blockchains: While in most cases (i.e., 13 out of 15 executions), more than 70% nodes hold the valid blockchain, only two executions exhibit slightly lower ratios (i.e., hardly below 70%). The bottom line is that, in practice, we need to guarantee at least 51% nodes' data are not tampered with. Besides, the experiment shows more than $\frac{2}{3}$ nodes remain intact, which is the case in a byzantine fault-tolerant system. Note that, by definition, a consensus is reached by having more than 50% of nodes holding valid blockchains. This experiment demonstrates that the real quorum in practice is far more than 51%. Therefore, in DEAN, the lower bound 51% could have been elevated. Since we are interested in only the safety of the protocol at this point, it is sufficient to show that 51% quorum is reached—finding out the minimal quorum (possibly higher than 51%) is beyond the scope of this paper. This result, however, suggests that we might be able to further trade the additional 17% quorum for even higher performance.

Another important phenomenon is that a larger portion (68%) of nodes hold the correct chain. We did not conduct any theoretical analysis on how many nodes could achieve unanimous agreement (i.e., 100% agreement); however, this experiment demonstrates that a significant portion (68% in this case) of nodes might not experience the so-called "divergence" problem—referring to that subsets of nodes (less than 51%) not holding the valid blockchain. Therefore, there might be some room in the DEAN protocol to further relax the constraints, possibly leading to higher performance.

C. Throughput

We report the throughput of the DEAN-based blockchain system prototype and compare its performance to other leading blockchain systems: Ethereum [16], Parity [17], Hyperledger [15], and IOTA [28]. We do not compare the RapidChain and Omniledger as the current sharding mechanisms are not appropriate for the IoT environment (see Section II). We measure the performance of DEAN on up to 32 nodes (the ratio of edge and sensor nodes is 1:3) over the period of five minutes, where each senor node issues up to 10,000 queries per second and each block contains more than 12 transactions. The workload is similar to [38] used for other blockchain systems [39].

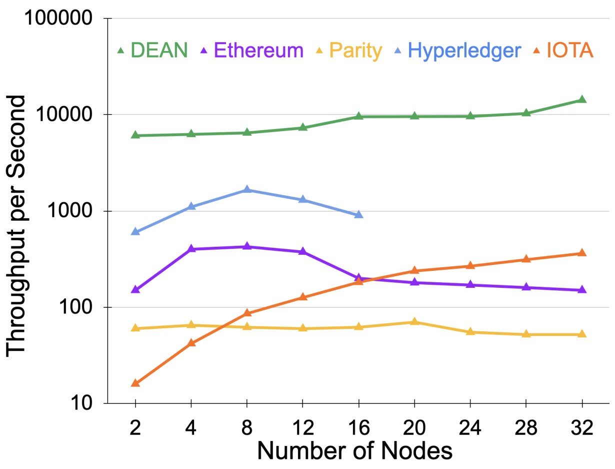


Fig. 8. **Throughput Comparison between DEAN and state-of-the-art blockchain systems.** IOTA is set with difficulty=1. DEAN outperforms major blockchain systems with orders of magnitude higher throughput.

Figure 8 reports that DEAN-based system outperforms all other systems in terms of the throughput. DEAN provides up to 88.6×, 16.6×, 6.7×, and 25× more throughput compared to Parity, Ethereum, Hyperledger, and IOTA respectively. DEAN exhibits significantly higher throughput because it could shorten the validation time without compromising the security guaranteed by the protocol, which exploits the unique property in edge computing. Hyperledger delivers much higher throughput than Parity because Hyperledger is not based on PoW, but relies on leader selection protocol (practical byzantine fault tolerance, PBFT), whose bottleneck lies on the network rather than the computing time. That is also why Hyperledger shows a poor scalability in literature [39] (usually not scalable beyond tens of nodes). DEAN, in contrast, not only achieves higher throughput than Hyperledger, but also scales the throughput beyond 16 nodes—the known limit for Hyperledger [39].

Parity delivers the lowest throughput among the five systems under comparison. Parity's consensus is based on a simplified version of Proof-of-Stake (PoS), which is obviously inappropriate to edge computing (see Section II) because it is difficult to estimate the "stake" of the edge and sensor nodes as they may come and go at arbitrary times. Although IOTA tends to exhibit improvement in throughput with the lowest difficulty (i.e., 1), in practice, the throughput is much lower due to higher difficulty, usually greater than 13 [40]. Performance-wise, this experiment suggests that DEAN is a preferable protocol for edge sensors and edge computing applications.

D. Latency

Similar to the previous section, we compare DEAN's latency with Ethereum, Parity, and Hyperledger, respectively. We do not compare IOTA here as it (i.e., IOTA) processes individual transactions rather than a block. The latency here in this context refers to the latency from the perspective of

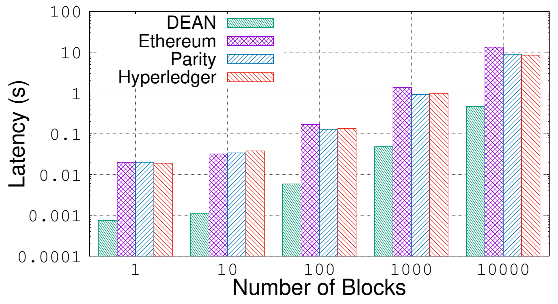


Fig. 9. **Latency comparison between DEAN and state-of-the-art blockchain systems.** DEAN is orders of magnitude faster than others and incurs only sub-second for adding 10,000 blocks.

TABLE III
CPU UTILIZATION AND MEMORY FOOTPRINT OF DEAN.

No. of Nodes	Total CPU Usage (%)	Memory per Node (MB)
4	3.7	12.23
8	8.5	12.61
12	13.8	12.65
16	18.1	12.66
32	33.8	12.83

the entire blockchain system rather than the conventional per-block latency. The latter is less interesting in this context because per-block performance is insufficient to fully describe the performance characteristic of the entire blockchain system, whereas the aggregated block-appending time represents the system latency as a whole.

As shown in Figure 9, DEAN incurs the lowest latency when appending various numbers of blocks ranging from one to 10,000. In particular, for appending 10,000 blocks, DEAN takes only 0.5 second while Ethereum takes 13 seconds, leading to 26× speedup; DEAN is also at least 10× faster than both Parity and Hyperledger.

It should be noted that in DEAN all the blocks contain more than 12 transactions, whereas each block generated in the experiments demonstrated at [39] contains only three transactions. Therefore, if we reduce the workload of DEAN, the latency gap would have been larger. In our current implementation of DEAN's system prototype, the number of transactions per block is hard-coded. Future releases will allow users to adjust the transaction density—the maximal number of transactions allowed in a single block (as long as size allows).

E. CPU and Memory Usage

We verify the resource consumption by DEAN at various scales on up to 32 nodes. To test the lightweight nature of DEAN, during this experiment, we use a single physical node equipped with a 32-core CPU and create all the edge nodes along with the sensors with Docker within the node with the default ratio (i.e., 1:3).

In Table III, we report the CPU usage rate by each specific cluster and the memory footprint by a node in that cluster. It is noticeable that at all scales, the memory footprint in each node remains within a reasonable limit (i.e., below 13 MB). The CPU usage rate is low (i.e., below 4%) at a smaller scale. Although the CPU usage rate is high at a large scale (i.e., 32 nodes), the overall usage is only about 1% per node in a specific core. This is because we distribute the 32 docker nodes within a physical node of 32 cores. Note that the goal of this experiment is to find out the maximum resource efficiency by DEAN within limited resources. The resource requirements in DEAN is fairly smaller compared to the current industrial specifications [41].

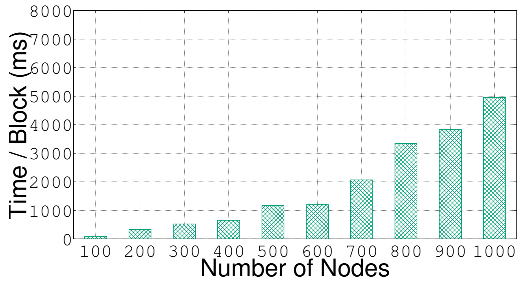


Fig. 10. **Scalability of DEAN's block processing time.** DEAN takes less than five seconds for processing a new block on extreme scales of 1,000 nodes.

F. Scalability

In this section, we report DEAN's performance at various scales on up to 1,000 nodes. To the best of our knowledge, little literature exists for the mainstream blockchain systems and protocols at such scales. In fact, it is well-accepted that existing PoW, PoS, and PBFT blockchain protocols hardly perform well at large scale [42]. For this reason, we did not compare DEAN to other blockchains in terms of scalability.

Figure 10 reports the time taken by DEAN for processing a single block at different scales ranging from 100 to 1,000 nodes. On 100 nodes, DEAN needs less than a quarter second for processing a block. On 1,000 nodes, the protocol can still deliver a new block within five seconds. Note that in our experimental setup each block comprises 12 transactions, implying that the overall system can process multiple transactions each second, which is on par, with the largest production blockchain system Bitcoin [14]. Indeed, Bitcoin consists of about 10,000 nodes globally [43], much more than the scale we are experimenting here. However, we argue that for edge computing applications where wireless networks are the norm and the performance bottleneck, 1,000 nodes are more than enough for edge computing applications to saturate the hardware resources as reported in [11]. While the next-generation high-performance wireless network, e.g., 5G network, is being widely deployed, the network's limitation will be somewhat relaxed and then the bottleneck will likely come back to the node scalability. For that reason, we believe further scaling the number of nodes will be an important research question to be addressed in our future work.

We also compare the block processing time with a recent IoT blockchain system [29] that operates on a very small scale (i.e., 50 nodes). We observe that at a 50-node scale, the protocol [29] requires almost 3 seconds, whereas, at a 100-node scale, our proposed DEAN protocol needs less than a quarter second. The reason behind the notable performance

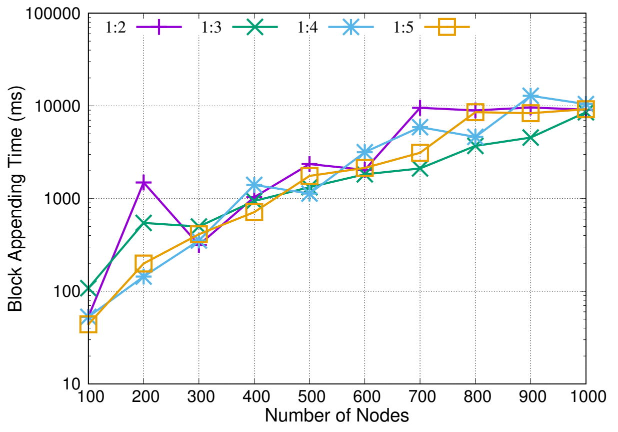


Fig. 11. **DEAN's block appending time with various distributions of sensor and edge nodes.**

degradation in [29] is, the protocol does not support parallel block processing and requires a pre-computation of block allocation before each block mining process that consumes a significant amount of processing time. In contrast, the block dissemination mechanism proposed in DEAN is independent of the mining process.

G. Sensitivity with increasing workload

All experiments thus far discussed assume the ratio between sensor nodes and edge nodes is 1:3, where each edge node is connected by three sensor nodes and all edge nodes are connected with a full mesh network. It is a natural question to ask how, if at all, the ratio would impact the performance of DEAN. The correctness of DEAN is out of the question regarding the ratios as long as the ratio M is larger than one. Therefore, the remainder of this section will evaluate the impact of different sensor:edge node ratios.

We vary the ratios between sensor and edge nodes as 1:2, 1:3, 1:4. and 1:5, and report the block appending time at different scales from 100 to 1,000 nodes. More sensor nodes generate more transaction requests. We change the default block size (i.e., 12 transactions) to 24 transactions to check the robustness. Ideally, the performance should be stable with little impact from the ratios—implying a strong stability of the proposed consensus and system implementation. Finding out the optimal ratio is part of the parameter tuning procedure and is beyond the scope of this paper. In addition, extreme large ratios (e.g., 1:100 or more) is not the norm of today's real systems [11] and thus will be investigated in our future work.

In Figure 11, we compare the block appending time incurred by different sensor:edge node ratios. Similarly to throughput, appending time does not seem impacted much by varying the ratios, and a smaller ratio exhibits more zigzags than larger ratios due to the same reasons discussed before: with more sensor nodes in the network, the entire system works more like a homogeneous system exhibiting a smoother curve in the performance plot. To summarize, from Figure 11, we can draw a conclusion that with the increment of transaction requests from sensors, the DEAN protocol guarantees consistent block processing with the growing workload.

V. Conclusion

This paper presents the DEAN consensus protocol to adopt blockchain in a resource-constrained environment like edge computing. The key idea of DEAN is partly enlightened by a scale-free graph **development** mechanism to find the most trusted nodes through an IoT-specific unique quorum election policy that assures the blockchain service through any uncertain state, and its safety is experimentally verified on a system prototype. In addition to the improved data fidelity, the system prototype also delivers orders of magnitudes higher performance than the state-of-the-art alternatives thanks to DEAN's unique design regarding the load balance on both computation and storage.

References

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.
[2] Brandon Haynes, Amrita Mazumdar, Armin Alaghi, Magdalena Balazinska, Luis Ceze, and Alvin Cheung. Lightdb: A dbms for virtual reality video. *Proc. VLDB Endow.*, 11(10):1192–1205, June 2018.
[3] H. Zhang and K. Zeng. Pairwise markov chain: A task scheduling strategy for privacy-preserving sift on edge. In *IEEE INFOCOM*, 2019.
[4] Anton Burtsev, David Johnson, Josh Kunz, Eric Eide, and Jacobus Van der Merwe. Capnet: security and least authority in a capability-enabled cloud. In *Proceedings of the 2017 Symposium on Cloud Computing*, pages 128–141, 2017.
[5] Bruno Medeiros, Marcos A Simplicio Jr, and Ewerton R Andrade. Multi-tenant isolation of what? building a secure tenant isolation architecture for cloud networks. In *Proceedings of the ACM Symposium on Cloud Computing*, pages 518–518, 2018.
[6] H. Zhou, X. Ouyang, Z. Ren, J. Su, C. de Laat, and Z. Zhao. A blockchain based witness model for trustworthy cloud service level agreement enforcement. In *IEEE INFOCOM*, 2019.
[7] Hackers Remotely Kill a Jeep on the Highway. https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway, Accessed 2019.
[8] Report: 97% of mobile malware is on android. This is the easy way you stay safe. shorturl.at/rvxF1, Accessed 2019.
[9] Jin Li, Lichao Sun, Qiben Yan, Zhiqiang Li, Witawas Srisa-an, and Heng Ye. Significant permission identification for machine-learning-based android malware detection. *IEEE Transactions on Industrial Informatics*, 14(7):3216–3225, 2018.
[10] J. Chen, S. Yao, Q. Yuan, K. He, S. Ji, and R. Du. Certchain: Public and efficient certificate audit based on blockchain for tls connections. In *IEEE INFOCOM*, 2018.
[11] Jiayao Wang, Abdullah Al-Mamun, Tonglin Li, Linhua Jiang, and Dongfang Zhao. Toward performant and energy-efficient queries in three-tier wireless sensor networks. In *ICPP*, 2018.
[12] Leslie Lamport. Paxos made simple, fast, and byzantine. In *OPODIS 2002, Reims, France, December 11-13, 2002*, pages 7–9, 2002.
[13] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, November 2002.
[14] Bitcoin. https://bitcoin.org/bitcoin.pdf, Accessed 2018.
[15] Hyperledger. https://www.hyperledger.org/, Accessed 2020.
[16] Ethereum. https://www.ethereum.org/, Accessed 2020.
[17] Parity. https://www.parity.io/, Accessed 2020.
[18] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 583–598. IEEE, 2018.
[19] Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. Rapidchain: Scaling blockchain via full sharding. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2018.
[20] Hyperledger. https://github.com/iotaledger/iota.go, Accessed 2020.
[21] GM and BMW Back Blockchain for Self-driving Cars. https://www.coindesk.com/gm-bmw-back-blockchain-data-sharing-for-self-driving-cars, Accessed 2019.
[22] Haoting Shen, Shahriar Badsha, and Dongfang Zhao. Consortium blockchain for the assurance of supply chain security. In *Network and Distributed System Security Symposium (NDSS)*, 2020.
[23] U.S. Department of Homeland Security. Combating trafficking in counterfeit and pirated goods - report to the president of the united states, 2020.

[24] Institute for Defense Analyses. Supply chain risk in leading-edge integrated circuits, 2021.
[25] Matthew Areno. Supply chain threats against integrated circuits, 2020.
[26] Daniel DiMase, Zachary A Collier, Jinae Carlson, Robin B Gray Jr, and Igor Linkov. Traceability and risk analysis strategies for addressing counterfeit electronics in supply chains for complex systems. *Risk Analysis*, 36(10):1834–1843, 2016.
[27] Cybersecurity and Infrastructure Security Agency. Understanding Denial-of-Service Attacks, 2019.
[28] Serguei Popov. The tangle. *White paper*, 1(3), 2018.
[29] Yaodong Huang, Jiarui Zhang, Jun Duan, Bin Xiao, Fan Ye, and Yuanyuan Yang. Resource allocation and consensus on edge blockchain in pervasive edge computing environments. In *IEEE ICDCS 2019*, pages 1476–1486, 2019.
[30] Laphou Lao, Xiaohai Dai, Bin Xiao, and Songtao Guo. G-pbft: a location-based and scalable consensus protocol for iot-blockchain applications. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 664–673. IEEE, 2020.
[31] Jiaping Wang and Hao Wang. Monoxide: Scale out blockchains with asynchronous consensus zones. In *NSDI*, pages 95–112, 2019.
[32] The Small-World Phenomenon: An Algorithmic Perspective. https://www.cs.cornell.edu/home/kleinber/swn.d/swn.html, Accessed 2021.
[33] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32:4868–4879, 2019.
[34] Sergey N Dorogovtsev, Alexander V Goltsev, and José Ferreira F Mendes. Pseudofractal scale-free web. *Physical review E*, 65(6):066122, 2002.
[35] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. Towards scaling blockchain systems via sharding. In *SIGMOD*, pages 123–140, 2019.
[36] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, USA, 1st edition, 2009.
[37] Docker. https://github.com/docker/docker, Accessed July 16, 2019.
[38] YCSB. https://github.com/brianfrankcooper/YCSB/wiki/Core-Workloads, Accessed 2018.
[39] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. Blockbench: A framework for analyzing private blockchains. In *ACM SIGMOD*, 2017.
[40] Umair Sarfraz, Sherali Zeadally, and Masoom Alam. Outsourcing iota proof-of-work to volunteer public devices. *Security and Privacy*, 3(2):e98, 2020.
[41] Raspberry Pi. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/, Accessed 2021.
[42] Kaiwen Zhang and Hans-Arno Jacobsen. Towards dependable, scalable, and pervasive distributed ledgers with blockchains. In *IEEE ICDCS*, 2018.
[43] Bitcoin Scale. https://bitnodes.earn.com, Accessed 2019.