Blockchain in IoT Systems: End-to-End Delay Evaluation

Maha Alaslani, Faisal Nawab, and Basem Shihada, *Senior Member, IEEE*

***Abstract*—Providing security and privacy for the Internet of Things (IoT) applications while ensuring a minimum level of performance requirements is an open research challenge. Recently, blockchain offers a promising solution to overcome the current peer-to-peer networks limitations. In the context of IoT, Byzantine fault tolerance (BFT)-based consensus protocols are used due to the energy efficiency advantage over other consensus protocols. The consensus process in BFT is done by electing a group of authenticated nodes. The elected nodes will be responsible for ensuring the data blocks' integrity through defining a total order on the blocks and preventing the concurrently appended blocks from containing conflicting data. However, the blockchain consensus layer contributes the most performance overhead. Therefore, a performance study needs to be conducted especially for the IoT applications that are subject to maximum delay constraints. In this paper, we obtain a mathematical expression to calculate the end-to-end delay with different network configurations, i.e., number of network hops and replica machines. We validate the proposed analytical model with simulation. Our results show that the unique characteristics of IoT traffic have an undeniable impact on the end-to-end delay requirement.**

***Index Terms*—Blockchain, Byzantine consensus, end-to-end delay, Internet of Things (IoT), queueing, realtime.**

I. Introduction

INTERNET of Things (IoT) is a paradigm that ranges from small, localized systems to large, geographically distributed systems that interconnect things to the Internet by using standard communication protocols [1]. IoT received massive attention from numerous business and technological industries that made the IoT one of the most demanded technologies of the future. However, modern IoT systems come with stringent network delay needs. Apart from this, the existing IoT systems are cloud centered. And, sending the data all the way to the cloud servers can easily break the delay requirements. Despite centralization and controlled data, cloud-supported IoT devices are not safe from cyber-crime, privacy issues, and security breaches. It is a fact that the single point of failure and the security flaws in IoT devices have placed data integrity and privacy at risk. It is shown that more than 25% of corporate attacks and cyber-crimes would be because of the compromised IoT devices [2]. It has become mandatory that the operational model of IoT devices should be shifted from over-arched centralized model to automated decentralized architecture. This transformation will help to make the IoT devices more self-regulating and autonomous.

Providing QoS assurance for the IoT real-time and mission-critical services over the current network infrastructures has been a major research challenge. Applications such as smart transportation system, smart industry, and E-health services are highly time-sensitive and require QoS guarantees. Data loss and latency may affect the accuracy and completion time of any task, and such quality measurements must be incorporated into any system specifications. We observe that blockchain technology offers a promising solution which can be helpful to provide the needed infrastructure for the emerging IoT applications. As the current infrastructure hardly meets the IoT requirements and most of the strict needs cannot be fulfilled in today's configurations. For example, in a smart factory, motion control systems require the network delay to be between 0.5 and 2 ms [3]. By definition, blockchain is a decentralized technology in which data is secured in the form of cryptographically linked blocks. Once the data is recorded on the blockchain ledger, it is extremely difficult to remove it [4]. By providing a secure distributed network, blockchain can deliver a platform for IoT to interconnect reliably and avoid privacy and security threats of central cloud models. The statistic from [2], shows that about 51% of global blockchain use cases were focusing on IoT. However, blockchain technology is still suffering from several challenges including scalability, and performance limitations. Through defining a total order of appended blocks and preventing concurrently appended blocks from containing conflicting data, the data integrity can be guaranteed. Thus, consensus protocols in blockchain are one of the most important and revolutionary aspects that allow a blockchain to be updated, while preserving data blocks' coherence.

This paper aims at analyzing the network delay for a blockchain system that serves massive and critical IoT applications. As shown in Fig. 1, our system is composed of multiple IoT devices with different requirements. The data gathered from these devices will be sent to the consensus cloud to be verified and appended to the blockchain. Incorporation of blockchain technology in an IoT system maximizes its security by creating an additional security layer. In the blockchain setting, the consensus layer is the responsible feature for ensuring that every block in the chain is true and authentic.

Manuscript received March 11, 2019; revised April 29, 2019; accepted May 7, 2019. Date of publication May 16, 2019; date of current version October 8, 2019. *(Corresponding author: Basem Shihada.)*

M. Alaslani and B. Shihada are with the Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia (e-mail: maha.aslani@kaust.edu.sa; basem.shihada@kaust.edu.sa).

F. Nawab is with the Department of Computer Science, University of California at Santa Cruz, Santa Cruz, CA 95064 USA (e-mail: fnawab@ucsc.edu).

Digital Object Identifier 10.1109/JIOT.2019.2917226

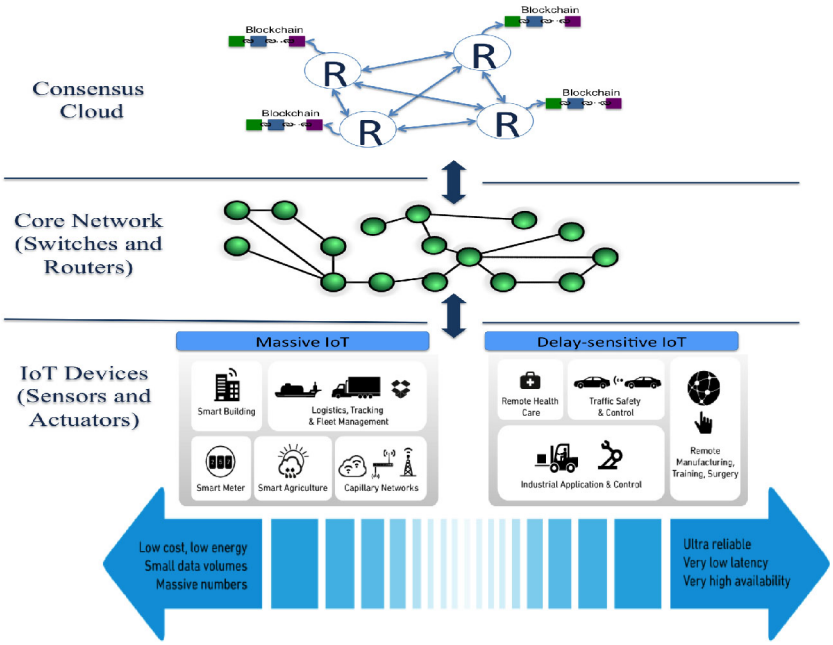

Fig. 1. General architecture of Byzantine-based IoT blockchain system [5].

Generally, consensus protocols in blockchain can be further categorized into two types: 1) proof-based and 2) BFT-based consensus [4]. The proof-based blockchain is about the selection of a random node to append a block to the chain. However, this method offers probabilistic guarantees that not safe when two nodes are selected to append the same block in the chain. Moreover, the consensus mechanism in the proof-based category is called mining. Mining is a complex process and computer with highly specialized hardware is needed. This process consumes large amounts of power and this huge costs can be claimed as the major drawback and prevented its effectiveness on the resource constraints IoT devices. The second category is the BFT-based consensus. In this category and before verifying a new block and making the final decision, the data are exchanged among a small group of authenticated nodes known as, replicas. The authentication process and the energy efficiency are the two main advantages for BFT-based consensus compared to the proof-based mechanisms.

However, the consensus layer contributes the most delay overhead. The current performance of blockchain technology is incompatible with the majority of IoT applications. Latency limitation is a primary cause of this incompatibility and an improvement is needed to overcome this limitation. Thus, in IoT networks Byzantine fault tolerant (BFT) family of protocols is used as consensus mechanism. A Byzantine fault is a fault that intentionally destroys a system operation and such failures are hard to detect by failure detection systems [6], [7]. Practical BFT (PBFT) [8] is one well-known member of the BFT family of protocols, PBFT can tolerate f malicious or Byzantine failures with $3f + 1$ replica nodes and the block is confirmed immediately when it is appended to the chain. From the moment that an IoT device sending its data to the consensus cloud until the data is confirmed and appended to the blockchain, a number of network hops and consensus machines (replicas) are used. However, adding more machines to the consensus process will considerably improve its reliability but also increases the delay. Thus, we obtain a mathematical expression to calculate the end-to-end delay with different network configurations (number of network hops and replica machines) to guarantee a minimum performance for the IoT applications. We evaluate our model by using simulation with two IoT applications (smart industry [3] and smart city [5]). Our evaluation demonstrates the huge overhead added by the network infrastructure and the indubitable impact of the unique characteristics of IoT traffic. Our results show that the high-level application requirements must be considered in the early design stages for the sake of meeting the performance expectations.

To summarize, the key contributions of this paper are as follows.

1) The network end-to-end delay of IoT applications is calculated.
2) The network hop-counts and the number of the consensus machines are considered as the key configuration parameters that directly affects the IoT applications performance.
3) The Usage of IoT use cases with different predefined requirements and broad range of traffic characteristics, including packet arrival rate, payload size, devises density, and surface area of deployment, to validate our proposed mathematical model.
4) The evaluation over various network setups and application requirements.

II. Motivation and State-of-the-Art

The IoT system is composed of different heterogeneous devices interconnected to each other to collect different kinds of information. It is an incredibly diverse space, encompassing a large variety of applications and services with

TABLE I
LATENCY REQUIREMENTS FOR DIFFERENT SMART FACTORY'S USE CASES [3]

Use Case	Priority	Latency (ms)	Average Payload Size	Device Density ($100\ m^2$)
Motion Control	Very high	0.5 - 2.0	63 bytes	185
Mobile Control	High	4.0 - 12.0	135 bytes	22
Process Monitoring	Medium	<50.0	60 bytes	1000
Video Remote Control	Low	10.0 -100.0	80 Kbytes	10

TABLE II
TRAFFIC CHARACTERISTICS OF MASSIVE IOT IN SMART CITY SCENARIO [5]

Use Case	Message Interval	Average Payload Size	Device Density (km^2)
Vending Machine	24 hours	150 bytes	150
Bike Management	30 minutes	150 bytes	200
Pay-as-You Drive	10 minutes	150 bytes	2250
Electricity Meters	24 hours	100 bytes	10000
Water Meters	12 hours	100 bytes	10000
Gas Meters	30 minutes	100 bytes	10000

different requirements and needs. Therefore, an overview of IoT application requirements is provided in this section. IoT applications can be subdivided into critical (delay-sensitive) and massive applications. Critical IoT applications are those which have high levels of delay and reliability requirements [5], common examples of such applications include traffic safety, automatic machinery and vehicles, artificial intelligence-based machines and remote surgery in the healthcare sector. On the other hand, massive applications require high accuracy, reliability, large number of connections, and small data volumes [5]. Weather monitoring, transport logistics, smart buildings, and smart metering are some examples of massive applications. Two detailed uses cases will be given as follows.

1) *Smart Industry (Industry 4.0):* Industry 4.0 [9] is the fourth stage of the industrial revolution. It is the next era of industrial production which is aimed to improve the flexibility and usability of future smart industries. Generally, Industry 4.0 is about the integration of IoT and its related services with the industrial manufacturing technologies. The industrial domain has very diverse requirements as it is comprised of a large number of heterogeneous use cases and applications. Among the important characteristics of different use cases that need to be considered are quality of services, cost-effectiveness, security, safety, reliability, and availability. The domain-specific personnel is responsible for considering these aspects with respect to their importance. Table I summarizes the latency requirements for different use cases in the industrial sector. Motion control [3] is one of the most challenging and demanding use cases among all the listed use cases. Basically, the motion control system deals with moving and rotating parts of a machine and dealing them effectively in a well-defined manner is a primary responsibility of this system. A use case with such responsibilities is expected to have very serious requirements of determinism, ultralow latency, and reliability.
2) *Smart Cities:* A smart city also works like the smart industry and it aims of improving the quality and effectiveness of government services for the sake of public welfare. Smart cities also use communication technologies to enhance operational efficiency and share the information with the common public. A network of connected devices is deployed in a particular area for better results such as electricity, water, and gas meters, vending machines, parking monitoring devices, and accelerometers in cars to keep an eye on car driver behavior. A dense area with 10 000 devices per each kilometer is deployed as a base for smart city services scenario, as the central areas of New York, London, and Beijing [5]. A most significant feature of these devices is the creation of nondeterministic behavior. This behavior is created through the wireless communication interface and the sharing of the same communication channels that introduces the contention on radio module and interference between the different sources of data traffic. With this contention, even minimum performance requirements cannot be guaranteed. Table II shows the traffic characteristics of IoT technology in a smart city scenario.

Instant consensus agreement is needed to satisfy the speedy transaction aspect of the above IoT systems. State-of-the-art consensus protocols can take from a few seconds up to several minutes to finalize the data on the blockchain. Proof of work (PoW) [10] consensus mechanism with different implementations such as Bitcoin [11], Litecoin [12], and Ethereum [13] has a finality time between 2.5 and 60 min [14]. While proof of stake (PoS) [15] in Ethereum Casper [16] and EOS [17] has a better finality ranging between 2 and 15 min [14]. On the other hand, consensus algorithms that follow the BFT principles, e.g., PBFT [8], delegated BFT (DBFT) [18], and delegated PoS (DPoS) with PBFT [19] can provide the block finality in the order of seconds [20]. For that, BFT-based consensus protocol, with some improvements, can be considered the most suitable candidate protocol that addresses this issue. As a proof of suitability, a list of some well-known IoT blockchain projects that implement BFT and its variants is introduced as follows.

1) *IBM Watson IoT Platform* [21] is specially designed for IoT devices that provide a managed, cloud-hosted blockchain, and analytic services. The data can be captured and explored to help the organizations achieve

their objectives. The consensus is provided by IBM Hyperledger Fabric that uses a BFT algorithm.

2) *IoT chain* [22] is a small operating system that built on the top of the blockchain concept and provides a proof-of-hardware security option. The large number of IoT nodes within the network will help to provide decentralized data protection. IoT chain uses PBFT to achieve main chain consensus.
3) *IoTeX* [19] consists of two blockchains, the root chain and the subchains that connected to form a single architecture for heterogeneous computing. To accelerate the transaction speed, IoTeX uses DPoS [23] and PBFT as the consensus mechanisms.
4) *NEO* [18], the smart economy is the main objective of NEO that can be activated by using the blockchain technology and the digital identity. Thus, the smart economy is the result of smart contracts, digital identity, and digital assets. Nodes on NEO use DBFT algorithm [18].

However, BFT has some limitations by being partially decentralized; the trust is placed in some known validator nodes. Moreover, these algorithms are believed to have high communication complexity and a careful performance study needs to be conducted. BFT-based consensus algorithm is widely studied in the literature. However, a limited number of works focused on the performance evaluation of BFT and its variants in the context of IoT networks and blockchains. Ripple, Hyperledger Fabric v0.6 with PBFT consensus, and Hyperledger Fabric, based on v1.0, with BFT-SMaRt [24] consensus were evaluated in [25]. The experimental results show that Byzantine consensus algorithms offer a reasonable throughput, but their performance does not scale to a large number of devices and drops dramatically as the number of participated devices increases. Moreover, the work in [26] provides an overview of the blockchain platforms used in the industrial IoT. The authors conclude that the current systems need to be modified in a way to meet the special needs of the IoT network. To the best of our knowledge, none of the previous works were interested in modeling the underlying network infrastructure of the Byzantine-based consensus blockchain system and analyzing the end-to-end requirements of the IoT applications.

III. Blockchain-IoT Integration Requirements

To support the strict requirements of the IoT applications, we need to revisit the consensus layer for the current blockchain implementations.

1) *Peer-to-Peer Distributed Ledger:* decentralization of network and verification of blockchain data can create many challenges and it is necessary to address these issues for the development of a robust, strong, useful, and accessible system. More precisely, the blockchain implementations currently process few transactions per second and the estimates show that number can go as high as 1500 transactions per second for Ripple blockchain system [27]. An individual record will take from a few minutes to many days to be confirmed [28], while IoT devices can generate up to 5 quintillion bytes of data every day (2.5 followed by 18 zeros) [29]. For that, latency becomes an obvious bottleneck for the mainstream adoption of blockchain technology in IoT scenarios.
2) *Byzantine-Based Consensus Layer:* This is based on state machine replication [8] where a service is replicated across different nodes in a distributed system. The service state and operations are maintained by each replica in the system. These replicas must be deterministic; a service execution must start in the same state and must always produce the same result given the same set of arguments. The deterministic behavior helps to ensure the safety property by guaranteeing that all non-faulty replicas agree on a total order for the execution of requests despite failures. This family of protocols can provide near-instantaneous finality on the inclusion of data into the blockchain. Yet, the network communication complexity with n consensus replicas is $O(n^2)$ messages per block [30]. This huge communication overhead needs to be further analyzed by exploring the network infrastructure, the network hop-counts and the number of consensus machines, to provide the needed performance guarantees
3) *Blockchain Models:* Current blockchain systems can be categorized into two models: a) permissionless and b) permissioned blockchain [4].
   a) *Permissionless (Public) Blockchain:* As the name indicates, permissionless blockchain networks are open to public and anyone can join them. The devices in these networks perform the computing functions required to verify the data transactions. On the participated devices, a copy of ledger is hosted where a distributed consensus mechanism finalizes these ledgers. Majority of blockchain transactions are recorded under pseudonyms. However, it is possible to track these transactions along the person identity who is carrying these transactions. This is why; a permissionless blockchain network cannot guarantee data privacy, and anonymity. Moreover, a permissionless blockchain is always at the risk of 51% attack [4]. Such attacks happen when malicious users have control over the half of compute power connected with the blockchain network. This attack, even with a short period of time, can result in complete destruction of data recorded on the blockchain network. Although 51% attacks happen rarely, they are considered as a real threat to the integrity of data recoded on these networks. Bitcoin and Ethereum are two common examples of permissionless blockchain network.
   b) *Permissioned (Private) Blockchain:* The primary element that differentiates the permissioned blockchain from permissionless blockchain is the presence of an access control layer that built into permissioned blockchain nodes. In permissioned blockchain, no one can join the network without permission where the participants have

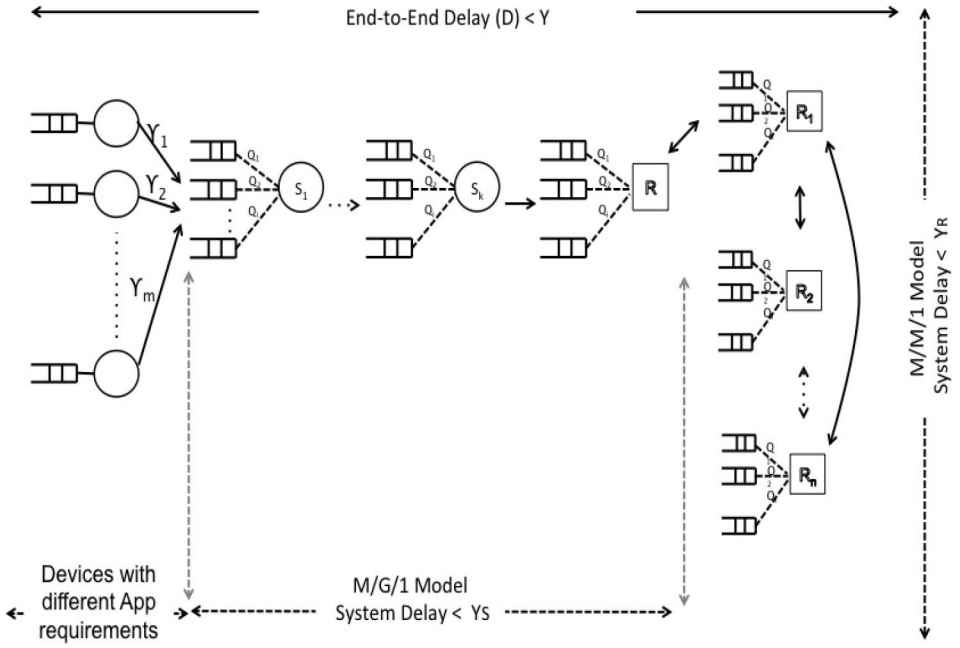

Fig. 2. Overview of the system.

complete control over the network. And, only approved parties are given full autonomy to participate in the consensus mechanism and validate the blocks of transactions. This step helps to mitigate the security challenges associated with permissionless blockchain. This is why; a permissioned blockchain network is more effective than permissionless in terms of data privacy. Moreover, permissioned blockchain that uses PBFT protocol can tolerate sybil attacks because of the membership aspect as long as the number of faulty nodes do not exceed one third of the total consensus machines. Sybil attack is a type of attack seen when a node attempts to gain inappropriate control over network peers by creating fake identities [31]. In addition, maximum data privacy is assured as outsiders are not allowed to access the transaction records. Due to the above advantages, permissioned networks are used by several large scale companies such as IBM [21].

For IoT, permissioned blockchain based on Byzantine consensus is a trustworthy and promising technology that can track a massive number of connected devices. However, permissioned blockchain model is comparatively more secure than other blockchain models because the identity verification is asked when a device joins the network. After verification, the new device is trusted and allowed to contribute information to the collective. As shown above, in case of performance permissioned blockchain network is more performant than the permissionless blockchain framework. In permissioned blockchain network, each node is dealing with a single application and it is required to perform the computation for that application. Thus, permissioned blockchain network keeps a balance between security, scalability, and performance. For the balance reason, permissioned blockchain model can be considered as the future of IoT infrastructure where security and trust are two key requirements of an IoT device to report the critical metrics.

IV. Byzantine-Based Blockchain System Model

Fig. 2 shows a high-level architectural view of our system. A blockchain system with a PBFT consensus protocol (PBFT) [8] will be modeled. The data will be collected from M IoT devices which belong to different application's use cases with different requirements. Then, the data will be broadcasted to the consensus sever (the leader) over K intermediate switches. Upon receipt the data at the consensus cloud, the replicas will communicate with each other to reach a consensus on the same data block (set of data records).

In the first part of the model, a single server facility with M/G/1 queue is considered where a maximum number of hops needs to be calculated to satisfy the application requirements.

1) Exponential interarrival times are used to model the IoT data packet arrival, with rates $\gamma_m, m = 1, 2, \ldots, M$.
2) The intermediate switches and the sink server are modeled as single server facilities with M/G/1 queue and multiple priority queues Q_p with a priority class $p \in \{1, 2, 3, \ldots\}$, where $p = 1$ is considered as the highest priority.
3) The intermediate switches and the sink server are assumed to operate following a general distributed service time of mean $\overline{X}_{S_i}$ for node $i \in \{1, 2, \ldots, K\}$, where each type of IoT traffic requires different service times.

TABLE III
NOTATIONS

Symbol	Definition
M	Number of IoT devices
N	Number of consensus replicas
Y	Predefined delay threshold for IoT applications
Y_S	Predefined delay threshold for a packet to reach the leader
Y_R	Predefined delay threshold to finish the consensus stage
λ	Packet arrival rate
$\overline{X}_{S_i}$	Mean service time of the network switches
$\overline{X}_{R_j}$	Mean service time of the replica machines
$\overline{Z}$	Mean residual time
ρ_p	The utilization for a priority class $p \in \{1, 2, 3, ...\}$
$\overline{W_p}$	Average waiting time of packets for a priority class $p \in \{1, 2, 3, ...\}$
$\overline{T_p}$	Average system delay for a priority class $p \in \{1, 2, 3, ...\}$
$\overline{L_p}$	Average number of packets for a priority class $p \in \{1, 2, 3, ...\}$
D	Total end-to-end delay
K^*	The maximum hop-counts
N^*	The maximum number of the replica machines
τ	The propagation delay

The second part of the model concerns about the number of consensus replicas R that needed to maintain the end-to-end requirements.

1) The internal communications between the replicas will be modeled as single server facilities with M/M/1 queue.
2) The replicas will be operated with an exponential service time distribution of mean $\overline{X}_{R_j}$ for a replica node $j \in \{1, 2, \ldots, N\}$.

The switches and the replicas are considered to be always running [32]. A summary of notations used throughout this paper is provided in Table III.

A. Delay Threshold Analysis

In order to maintain a guaranteed behavior for the IoT traffic, the expected delay to reach a consensus confirmation stage must not exceed a predefined threshold Y [33]. The predefined threshold can be seen as a part of a service level agreement (SLA) contract between the service provider and the end user [34]. In this paper, the predefined threshold Y has two main components: 1) a threshold value for a data packet to reach the sink node (consensus leader) Y_S and 2) a threshold value to finish the consensus confirmation stage Y_R. Therefore, a data packet with a cumulative delay exceeding Y will be treated by an appropriate decision. Thus, we define the total end-to-end delay as a random variable called D, then the probability for a data packet to violate delay requirement is given by

$$P_{\text{violating}} = \Pr\{D > (Y_S + Y_R)\}. \tag{1}$$

In such circumstances, our main intent is to calculate the acceptable hop-counts K^* and the number of the consensus replicas N^* such that the end-to-end delay $\mathbb{E}[D] < (Y_S + Y_R)$. Analyzing the application requirements especially the delay will help us to capture many of the inherent tradeoffs between the underlying network design and the reliability of the current blockchain system from one hand and the predefined requirements of the IoT applications at the other hand.

B. Network Hops Analysis

The network has a set of K switches and each switch has its own service rate distribution and, each switch is accommodated with multiple priority queues. The packet arrivals are assumed to have Poisson distribution, Poisson traffic is used to obtain closed-form results. Thus, we define the total arrival as

$$\lambda_i = \sum_{m=1}^{M} \gamma_m. \tag{2}$$

Each data packet will belong to a different priority class $p \in \{1, 2, 3, \ldots\}$ that will be experienced by sampling the packet at each relay node, the utilization can be expressed as the fraction of time an intermediate node S is serving a class p data packet, $\rho_{S_p} = \lambda_p \cdot \bar{X}_{S_p}$. The waiting time of a high-priority packet, denoted as W_{S_1}, is

$$W_{S_1} = \sum_{j=1}^{L_1} X_{S_j} + Z \tag{3}$$

where L_1 is the number of packets belongs to the high-priority class, X_{S_j} is the service time of packet j at node S, and Z is the residual time, the first moment of this residual time is

$$\overline{Z} = \frac{1}{2}\left(\sum_{i=1}^{p} \rho_{S_i} \cdot \frac{\overline{X_S^2}}{\overline{X_S}}\right). \tag{4}$$

By Little's formula, the high-priority packet average waiting time, denoted as $\overline{W}_{S_1}$, is

$$\overline{W}_{S_1} = \frac{\overline{Z}}{\left(1 - \rho_{S_1}\right)}. \tag{5}$$

The high-priority traffic has a second moment of waiting time as

$$\begin{aligned} \overline{W_{S_1}^2} &= \overline{L}_1 \cdot \mathrm{Var}(X_S) + \mathrm{Var}(Z) + \mathrm{Var}(L_1) \cdot \overline{X_S}^2 \\ &\approx \overline{L}_1 \cdot \mathrm{Var}(X_S) + \mathrm{Var}(Z) \end{aligned} \tag{6}$$

where

$$\begin{aligned} \overline{L}_1 &= \lambda_1 \, \overline{W}_{S_1} \\ \rho_{S_1} &= \lambda_1 \cdot \overline{X_S} \\ \mathrm{Var}(Z) &= \overline{Z^2} - \overline{Z}^2. \end{aligned}$$

Using the law of total expectation, the second moment of $\overline{Z^2}$ is

$$\overline{Z^2} = \sum_{i=1}^{p} \rho_{S_i} \cdot \overline{Z^2} = \frac{1}{3}\left(\sum_{i=1}^{p} \rho_{S_i} \frac{\overline{X_S^3}}{\overline{X}_S}\right). \tag{7}$$

From this, we understand that the waiting time for high priority packets is primarily due to residual time only and high priority queues contain at most one packet almost all the time. However, the waiting time of the lower-priority packets can be calculated as

$$W_{S_2} = \sum_{j=1}^{L_1} X_{S_j} + \sum_{j=1}^{L_2} X_{S_j} + \sum_{j=1}^{W_{S_2}\cdot\lambda_1} X_{S_j} + Z. \tag{8}$$

As per (8), a low-priority packet has a waiting time that can be expressed as the summation the service residual and the time that needed to serve existing high priority packets, new high priority packets, and existing low-priority packets that are ahead in the queue. Thus, a lower-priority packets have an average waiting time, denoted by $\overline{W}_{S_2}$ as

$$\overline{W}_{S_2} = \frac{\overline{Z}}{\left(1-\rho_{S_1}\right)\left(1-\rho_{S_1}-\rho_{S_2}\right)}. \tag{9}$$

The second moment of the average low-priority packet waiting time is

$$\begin{aligned}\overline{W_{S_2}^2} &= \overline{Z^2} + s\cdot \mathrm{Var}\left(X_{S_j}\right) + s^2\cdot \overline{X}_{S_j}^2 + 2\,s\,\overline{X}_{S_j}\cdot\overline{Z} + q\cdot\overline{X}_{S_j}^2 \\ &\approx \overline{Z^2} + s\cdot \mathrm{Var}\left(X_{S_j}\right) + s^2\cdot \overline{X}_{S_j}^2 + 2\,s\,\overline{X}_{S_j}\cdot\overline{Z}\end{aligned} \tag{10}$$

where the coefficient s is defined as

$$s = \overline{N}_2 + \overline{N}_1 + \lambda_1\cdot\overline{W}_{S_2}.$$

Here, q is the variance of the number of packets and the last approximation follows when it is assumed to be negligible in steady state.

In general, for a class p packets, the mean waiting time can be calculated using the same preceding approach as

$$\overline{W}_{S_p} = \frac{\overline{Z}}{\left(1-\rho_{S_1}-\rho_{S_2}-\cdots-\rho_{S_{p-1}}\right)\left(1-\rho_{S_1}-\rho_{S_2}-\cdots-\rho_{S_p}\right)} \tag{11}$$

where the total time a packet of priority class p_i spent in the system is

$$\overline{T_{S_p}} = \overline{W}_{S_p} + X_{S_p}. \tag{12}$$

Finally, the total number of packets in the system is given as

$$L_p = \lambda_p \overline{T_{S_p}}. \tag{13}$$

To this point, our objective is to obtain the maximum hop-count that respects the delay constraint

$$K^* = \arg\max_{K}\left\{\sum_{i=1}^{K}\overline{T}_{S_i} \le Y_S\right\} \tag{14}$$

where

$$\overline{T}_{S_i} = \overline{W}_{S_{p,i}} + \overline{X}_{S_i} + \tau_i \tag{15}$$

where $\overline{W}_{S_{p,i}}$ is given by (11) for each $i = 1, 2, \ldots, K$, $\overline{X}_{S_i}$ is node i service time, and τ_i is transmission delay between two consecutive nodes, node $i-1$ and node i.

Some exceptions may apply when all the nodes are indistinguishable, K^* can be simply calculated as

$$K^* \approx \frac{Y_S}{\overline{T}_S}. \tag{16}$$

C. Consensus Replicas Analysis

In our model, IoT devices across the network create and send the data packets to a leader machine at the consensus cloud that run the PBFT consensus protocol [8]. In the normal case PBFT runs a three-phase protocol, preprepare, prepare, and commit, to coordinate the consensus machines. In the first round, preprepare phase, the leader puts data packets together and forwards them to $N = 3f + 1$ replicas in the consensus network in the form of proposals. A proposal is an unconfirmed blockchain block that contains a batch of data records. Proposals are only forwarded when the leader has accumulated enough records, or a certain amount of time has elapsed since the last proposal. This prevents the leader from sending empty proposals. Then, each replica verifies the proposal (the block contents) and starts the next round by sending prepare messages. The third round is started when a replica received $2f+1$ approved prepare messages. In the third round, commit phase, a replica waits for $2f+1$ accepted commit before sending the result to the client, indicating that this block should be applied to the chain of each replica in the consensus network. Therefore, the client will wait for the results from at least $f+1$ replicas to accept the final result. More details are given in Algorithm 1. In our system, we have N replicas which are modeled as M/M/1 queue with exponential interarrival times with mean $1/\lambda_j$ and, exponential service times with mean $1/\mu_j$. The data is served in order of arrival and the traffic intensity is calculated as

$$\rho_{R_j} = \frac{\lambda_j}{\mu_j} \tag{17}$$

where j is the index that represents the replicas $j \in \{1, 2, \ldots, N\}$. The total time the traffic spend on a replica is given by

$$\overline{T}_{R_j} = \frac{1}{\mu_j\left(1-\rho_{R_j}\right)}. \tag{18}$$

We can derive the expression for the mean number of packets in the system by applying Little's law

$$L_{R_j} = \lambda\overline{T}_{R_j}. \tag{19}$$

The average waiting time can be calculated by subtracting the mean service time from (18) as

$$\overline{W}_{R_j} = \overline{T}_{R_j} - 1/\mu \tag{20}$$

or by applying Little's law, this yields the following:

$$\overline{W}_{R_j} = \frac{\rho_{R_j}/\mu}{1-\rho_{R_j}}. \tag{21}$$

Algorithm 1 PBFT

client c creates request m:
$o \leftarrow$ *state machine operation*
$t \leftarrow$ *timestamp*
$c \leftarrow$ *client id*
$\sigma_c \leftarrow$ *client signature*

upon reception of request $m =< REQUEST, o, t, c > \sigma_c$ $\wedge$ *replica_is_primary* **do**
$v \leftarrow$ *view*
$n \leftarrow$ *sequence number*
$m \leftarrow$ *client's request*
$d \leftarrow$ *digest of m*
$\sigma_p \leftarrow$ *primary signature*
multicast $<< PRE\_PREPARE, v, n, d > \sigma_p, m >$

upon reception of $<< PRE\_PREPARE, v, n, d >$ $\sigma_p, m >$ $\wedge$ *replica_is_backup* (i) **do**
$H \leftarrow$ *high water-mark*
$h \leftarrow$ *low water-mark*
if (σ_p is correct) $\wedge$ ($d = digest(m)$) $\wedge$ ($current\_view = v$) $\wedge$ (not accepted $d \neq digest(m) \parallel v = v \wedge n = n$) $\wedge$ ($h \leq n \leq H$) **then**
    accept $<< PRE\_PREPARE, v, n, d > \sigma_p, m >$
    multicast $< PREPARE, v, n, d, i > \sigma_i$
    $message\_log \leftarrow <<PRE\_PREPARE, v, n, d>\sigma_p, m>$
    $message\_log \leftarrow < PREPARE, v, n, d, i > \sigma_i$

upon reception of $< PREPARE, v, n, d, i > \sigma_i$ $\wedge$ (*replica_is_backup* $\vee$ *replica_is_primary*) **do**
$H \leftarrow$ *high water-mark*
$h \leftarrow$ *low water-mark*
if (σ_i is correct) $\wedge$ ($current\_view = v$) $\wedge$ ($h \leq n \leq H$) **then**
    accept $< PREPARE, v, n, d, i > \sigma_i$
    $message\_log \leftarrow < PREPARE, v, n, d, i > \sigma_i$

upon reception $(2f)$ of $< PREPARE, v, n, d, i > \sigma_i$ $\wedge$ (*replica_is_backup* $\vee$ *replica_is_primary*) **do**
$prepared(m, n, v, i) = true \iff$ *message_log* contains $m \wedge PRE\_PREPARE(v) = v \wedge PRE\_PREPARE(n) = n$
if $prepared(m, n, v, i) = true$ **then**
    multicast $< COMMIT, v, n, D(m), i > \sigma_i$

upon reception of $< COMMIT, v, n, D(m), i > \sigma_i$ $\wedge$ (*replica_is_backup* $\vee$ *replica_is_primary*) **do**
$H \leftarrow$ *high water-mark*
$h \leftarrow$ *low water-mark*
if (σ_i is correct) $\wedge$ ($current\_view = v$) $\wedge$ ($h \leq n \leq H$) **then**
    accept $< COMMIT, v, n, D(m), i > \sigma_i$
    $message\_log \leftarrow < COMMIT, v, n, D(m), i > \sigma_i$

upon reception $(f + 1)$ of $prepared(m, n, v, i) = true$ $\wedge$ (*replica_is_backup* $\vee$ *replica_is_primary*) **do**
$committed(m, n, v) = true$

upon reception $(2f + 1)$ of accepted *COMMIT* $\wedge$ $prepared(m, n, v, i) = true$ $\wedge$ (*replica_is_backup* $\vee$ *replica_is_primary*) **do**
$committed\_local(m, n, v, i) = true$
if $committed\_local(m, n, v, i) = true$ **then**
    executes the state machine operation (o)
    $r \leftarrow$ *result*
    send $< REPLY, v, t, c, i, r > \sigma_i$ to the client

upon client received $(f + 1)$ of $< REPLY, v, t, c, i, r > \sigma_i$ **do**
accept $< REPLY, v, t, c, i, r > \sigma_i$

In order to obtain the allowable number of the replica machines that meet our defined delay threshold, our objective here is similar to (14)

$$N^* = \arg\max_N \left\{ \sum_{j=1}^{N} \overline{T}_{R_j} \leq Y_R \right\}. \tag{22}$$

We know that, the commands are executed in deterministic order across all replicas and thus,

$$N^* \approx \frac{Y_R}{\overline{\overline{T}}_R}. \tag{23}$$

In another word, the total delay D experienced by the system should be

$$D \leq \frac{Y}{K^* + N^*} \tag{24}$$

or alternatively as

$$D \leq \sum_{i=1}^{K} \overline{T}_{S_i} + \sum_{j=1}^{N} \overline{T}_{R_j}. \tag{25}$$

V. Performance Evaluation

In this section, the performance of the above framework is evaluated using MATLAB/Simulink package. We are evaluating an abstract model and the details of the underlying infrastructure and technologies are ignored. The general IoT network topology for conducting our experiments is shown in Fig. 2 and the simulation parameters in Table IV. Noting that, replicas service time, $X_R = 0.4$ ms [35], encapsulate the complexity of PBFT three-phase processing.

A. Evaluation Setup

We did an extensive evaluation of our model using two IoT applications, smart factory and smart city, as per Section II. Smart factory or Industry 4.0 can be considered as delay-sensitive IoT application where strict latency requirements should be fulfilled. Four smart factory's use cases are considered in our experiments. As shown in Table I [3], the delay requirement varies between 0.5 and 100 ms. By considering these requirements, we define the end-to-end delay threshold Y to be 2, 12, 50, and 100 ms for the motion control, mobile control, process monitoring, and video control use cases, respectively. Also, we consider average size

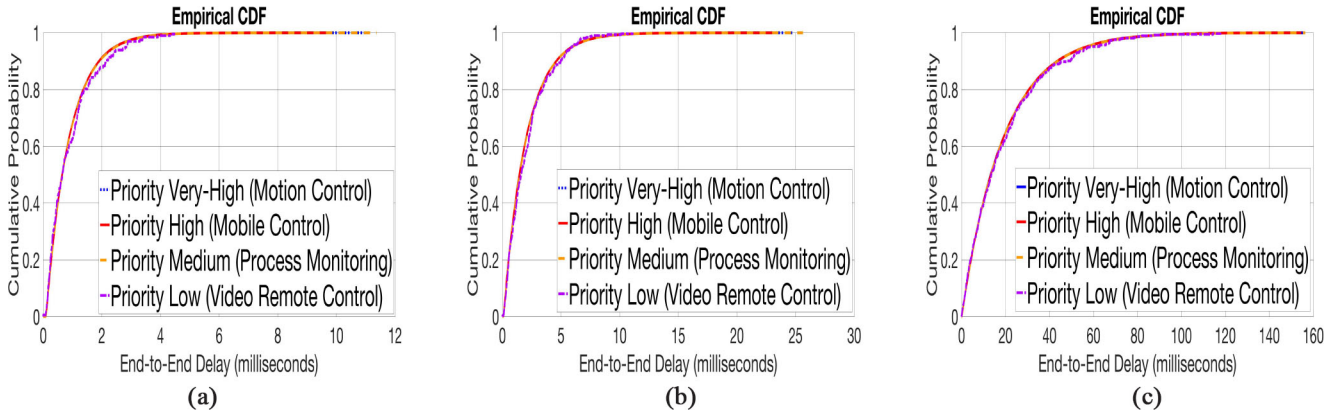

Fig. 3. End-to-end delay CDF for average size factories. (a) 40 000 sq. ft. (b) 65 000 sq. ft. (c) 100 000 sq. ft.

TABLE IV
SIMULATION PARAMETERS

Parameter	Value	
Simulation time	6000 sec	
Switches service time (X_S)	100 μsec [36]	
Replicas service time (X_R)	0.4 msec [35]	
Propagation delay (τ)	8.2 nsec/meter [36]	
Number of network hops (K)	1,5,15,30,45,60	
Number of replicas (N)	4,7,10,13	
Smart Factory (λ) [3]	Motion Control	packet/min * 64 bytes
	Mobile Control	packet/min * 135 bytes
	Process Monitoring	packet/min * 60 bytes
	Video Control	packet/min * 80 Kbytes
Smart City (λ) [5]	Vending Machine	packet/ 24 hr
	Bike Management	packet/ 30 min
	Pay-as-You Drive	packet/ 10 min
	Electricity Meters	packet/ 24 hr
	Water Meters	packet/ 12 hr
	Gas Meters	packet/ 30 min

factories [37], 40 000, 65 000, and 100 000 sq. ft where the density of the IoT devices is given in Table I [3]. However, the smart city's uses cases are examples of massive IoT. In such application, we relax our delay threshold Y to be 100 s [5] and any packet arrived after this time will be useless and action needs to be taken. Devices' density per km^2 and details packet arrival rates are given in Table II [5]. In our experiments, three cities scenarios with 300, 600, and 900 km^2 are evaluated; these values are obtained according to average sized cities in the USA [38].

B. Evaluation Metrics

In our evaluation we focus on the end-to-end delay, two metrics are used for this assessment.

1) *CDF* express the probability that if a defined variable takes a value less than or equal to x [39]. Our results show CDF for the end-to-end delay D

$$F(x) = \Pr[D \leq x] = \alpha.$$

2) *Percent deviation* is defined as the difference between the mean of a set of data from a theoretical or predefined value. This can be useful in our experiments to show the delay deviation from the applications defined metrics. We study the percent deviation from the delay threshold Y

$$\text{PercentDeviation} = \frac{D - Y}{Y} * 100\%.$$

The negative value signifies that the given mean of the delay is lower than the threshold Y. If the percent deviation is positive, it signifies that the delay mean value is higher than expected. As described above, we want to study the direct effect of the network hop-counts and the number of replica machines involved in the consensus protocol on the delay-sensitive and massive IoT applications.

C. Results and Discussion

End-to-end delay CDF for different smart factories are shown in Fig. 3. In Fig. 3(a), with 40 000 sq. ft, 90% of the packets encounter delay of less than 2 ms. On the other hand, 90% of the packets have a delay of less than 5 ms for the 65 000 sq. ft as shown in Fig. 3(b). While at 100 000 sq. ft, in Fig. 3(c), the four uses cases incur a delay of less than 40 ms for 90% of the traffic.

The average end-to-end delay percent deviation results for multiple hops and four replica machines are shown in Fig. 4. As shown in Fig. 4(a), it is hard to meet the requirement of the motion control use case where the delay should be less than Y of 2 ms. With less devices' density in the case of 40 000 sq. ft, the average delay for different hop-counts is between [+61%, +393%] percent deviation from the Y of 2 ms. As presented, when the devices' density for both 65 000 and 100 000 sq. ft increase the percent deviation is escalated up to the range of [+296%, +658%] and [+4.1e+3%, +4.6e+3%].

Mobile control yields better delay performance, Fig. 4(b), Y of 8 ms is fulfilled for 40 000 sq. ft with maximum hop-counts of 30. In 65 000 sq. ft, Y can be satisfied with less than five hops and it is impossible for the devices at 100 000 sq. ft to meet their application requirements.

In the third use case in Fig. 4(c), process monitoring meets its predefined Y of 50 ms for both 40 000 and 65 000 sq. ft with different hop-counts. For the 100 000 sq. ft, a percent

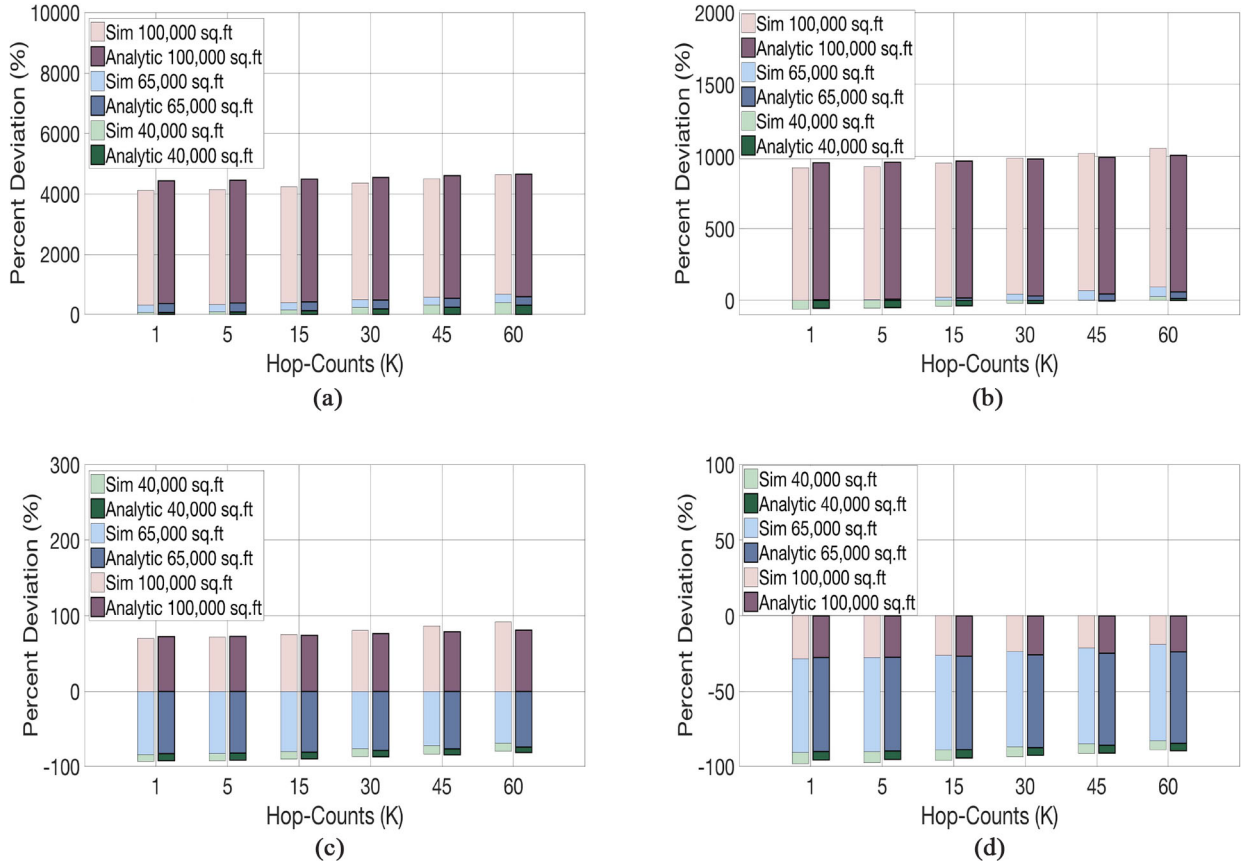

Fig. 4. Percent deviation from the average delay for different hop-counts (K) and four replica machines ($N = 4$). (a) Motion control ($Y = 2$ ms). (b) Mobile control ($Y = 8$ ms). (c) Process monitoring ($Y = 50$ ms). (d) Video control ($Y = 100$ ms).

deviation of $[+71\%, +91\%]$ is come across different hop-count configurations.

The video remote control with $Y = 100$ ms has the least restrictive delay requirement. This requirement is satisfied at the different network sizes and hop-counts, as illustrated in Fig. 4(d).

In the second experiment, we vary the number of replicas ($N = 3f+1$) from 4, 7, and 10, to 13 to tolerate 1–4 Byzantine faults f as shown in Fig. 5 and we obtain the results for each use case. Fig. 5(a) shows that it is impossible to meet the strict requirement when $Y = 2$ even with the different network configurations. In Fig. 5(b) and with number of replicas less than 7, we can meet the mobile control ($Y = 8$) requirement for the area of 40 000 sq. ft and less than four replicas for 65 000 sq. ft.

However, the different replica configurations, in Fig. 5(c), can meet the process monitoring ($Y = 50$) requirement for a surface area of 40 000 and 65 000 sq. ft but not for the 100 000 sq. ft. For the 100 000 sq. ft, the delay is in order of seconds due to the contention between the huge number of devices which introduce an obstacle to meet the application's delay requirement. As shown in Fig. 5(d), the delay requirement of the video control case ($Y = 100$) and four replica machines is satisfied.

In a smart city, the huge amount of participated devices is one of the unique characteristics of IoT systems that negatively impact the usability of the currently deployed applications. For this, we extend our evaluation and test a smart city scenario with six different use cases. The end-to-end latency constraint Y is relaxed to be 100 s [5]. First of all, CDF results in Fig. 6 are obtained. As shown, 90% of the packets for all the cases have a delay less than $10E6$ ms at an area of 300 km^2. For 600 km^2, only 30% of the packets have the $10E6$-ms delay. However in 900 km^2, 20% of the packets have the $10E6$-ms delay. As known the congestion occurs when the number of packets being transmitted through the network approaches the packet handling capacity of the network and the network delay will increase with the congestion. In this scenario, the average delay is always higher than the predefined requirement (100 s). Our main intend here to conclude that the current BFT-based blockchain cannot handle the huge amount of data generated by the different IoT applications.

VI. Open Issues and Future Works

BFT-based consensus family is one of the best choices with promising results for blockchain implementation in IoT system. However, several issues and challenges need clarifications and investigations to provide a solid infrastructure that meets the application-level requirements and end-user expectations.

1) *Data Finality Time:* As illustrated in this paper, the end-to-end delay of the current consensus algorithms, from

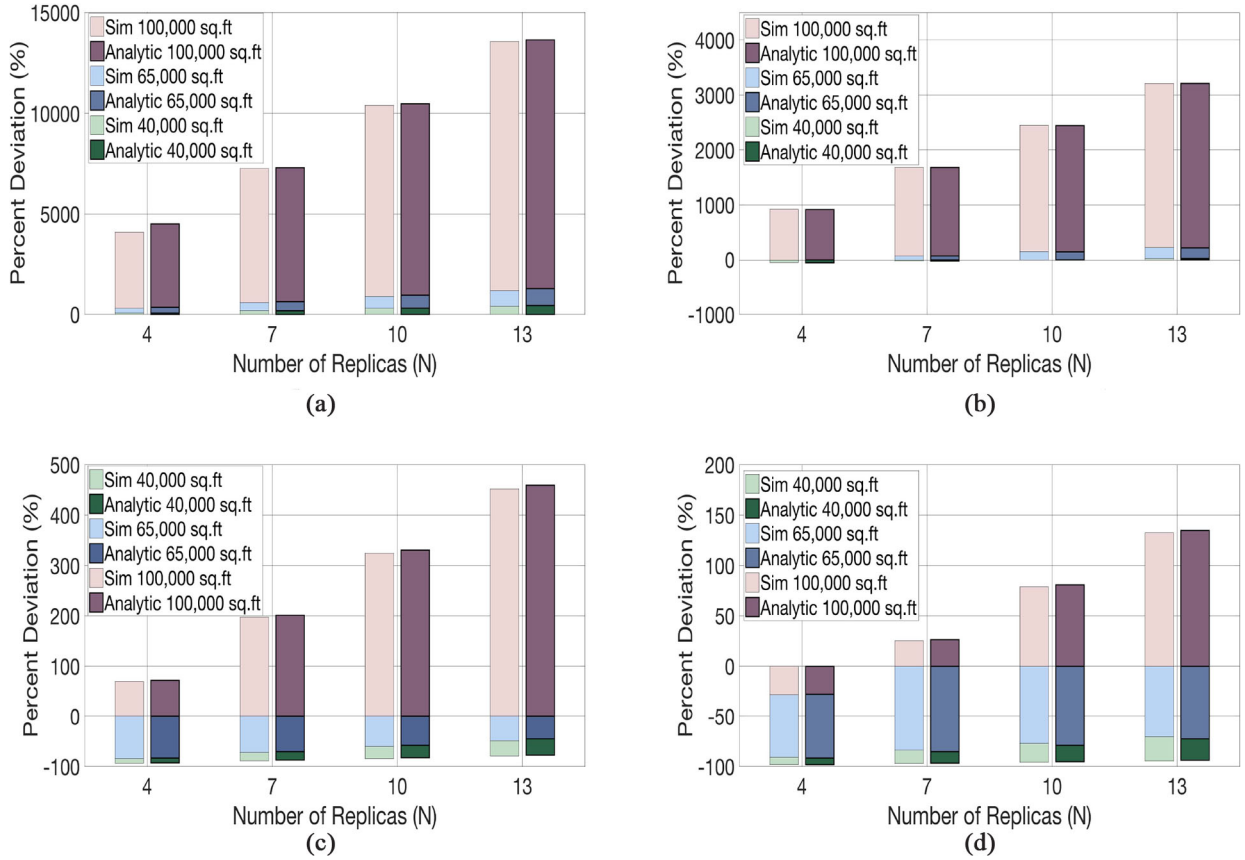

Fig. 5. Percent deviation from the average delay for different number of replica machines (N) and one-hop ($K = 1$). (a) Motion control ($Y = 2$ ms). (b) Mobile control ($Y = 8$ ms). (c) Process monitoring ($Y = 50$ ms). (d) Video control ($Y = 100$ ms).

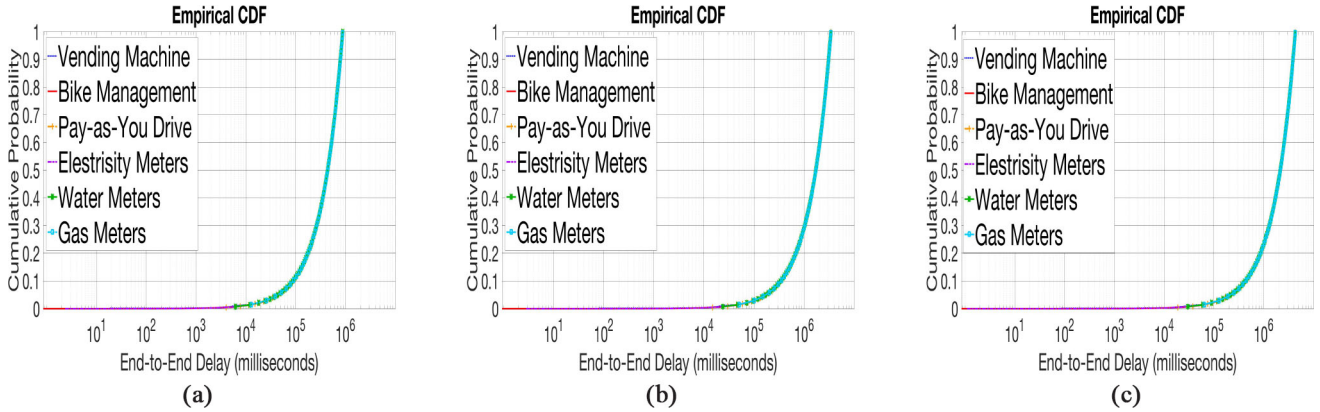

Fig. 6. End-to-end delay CDF for average size cities. (a) 300 km^2. (b) 600 km^2. (c) 900 km^2.

the time when the data is generated until its final enclosure in the blockchain, needs further improvements. Thus, it is essential to find a mechanism that maintains a balance between the delay performance measurements and the decentralization advantages of the blockchain.

2) *Space Complexity:* Typical blockchain has its own problems in regards to data storage. These problems require new protocols to be integrated on top of existing blockchains. As an example, Bitcoin reaches approximately 210 GB in size [40] and this high level of growth is incomparable with the large volumes of data generated from IoT devices. Storage mechanisms will be considered as one greatest challenge to be addressed at both research and industrial domains.

3) *Security, Access Control, and Trust Management:* With the huge amount of participated devices, define the access roles and privileges management in a distributed nature without a central authority is another research problem. IoT devices come with different capabilities and characteristics. Therefore, authentication techniques and built-in cryptographic primitives need to consider the limited resources drawback.

4) *Data Validity and Accuracy:* From a data standpoint, wireless communication could lead to erroneous or altered data that could compromise the validity of the data coming over the network. The system cannot ensure the accuracy of the data unless it can be secured through the generation phase. This is a challenge as security embedded at the IoT devices is often the most difficult to implement and maintain especially, in the case of resource-constraints sensors.

Improvements to the semidistributed BFT algorithms by partitioning the large-scale IoT network and reducing the communication complexity is one solution to the wide adoption of blockchain in IoT context.

1) *BFT-Based Consensus Optimization:* BFT-based consensus protocols are not truly decentralized where the trust is placed in some predefined nodes. Thus, fog computing model can be considered as a potential solution instead of using the cloud model. However, devices capabilities can raise another challenge that needs to be addressed. The second optimization that needs to deal with is the high number of exchanged messages between the replica machines (control packets). It is important to reduce the number of control packets to reduce the congestion level, the IoT data packets latency, and loss.
2) *"Big Data" Management Solutions:* The improvements can be done by integrating the big data management solutions. For instance, applying data mining solutions to look into the patterns and relationships of the IoT data, then using the machine learning mechanisms to improve the information usability, reduces the noise, and/or the redundancy, and finally feed the valuable data into the blockchain.
3) *Emerging Networking Architectures:* Another proposed solution can be done by combining the emerging networking architectures such as software-defined-networking (SDN) with blockchain. Intelligent matching and routing of the data based on dynamically applied rules that fed into the network by the applications can help to partition the network and reduce the huge amount of redundant data.

These proposed solutions are based on hypotheses and needed to be implemented and tested to prove its usability and applicability.

VII. Conclusion

In this paper, we studied the impact of the emerging IoT applications on the end-to-end delay incurred by Byzantine-based blockchain systems. An analytical model was driven and, the results were supported using MATLAB simulations. The results showed that the importance of incorporating the unique IoT characteristics in designing Byzantine-based blockchain for IoT networks. Network-level configuration, hop-counts, the number of replicated machines, and its relation to the devises density and variability were analyzed. The high contention between the packets masked the delay introduced by the hop-counts, this eliminated the data packets from meeting the application requirements even with different hops configuration. On the other hand, hop-counts directly affected the delay performance in an environment that has less contention between its devices. Moreover, It is well-know that adding more replica machines will add more delay, but it will increase the system reliability. Thus, the design decision should be taken according to the traffic characteristics and the application level requirements. The impacts of a broad range of IoT traffic characteristics including packet arrival rate, payload size, devises density, and surface area of deployment on Byzantine-based consensus were studied. The contention between IoT devices, high level of noise and redundancy in data created a management challenge for the network operators. Sending this large-scale data to the consensus cloud as existing IoT-based blockchain system will introduce another network and bandwidth bottleneck. Thus, design proposals that help to speed up data processing, and improve data adaptability and extensibility should be considered.

References

[1] R. Minerva, A. Biru, and D. Rotondi, "Towards a definition of the Internet of Things (IoT)," *IEEE Internet Initiative*, vol. 1, no. 1, pp. 1–86, May 2015.

[2] *Statista—The Portal for Statistics*. Accessed: Nov. 25, 2018. [Online]. Available: https://www.statista.com/

[3] "5G for connected industries and automation," Frankfurt, Germany, 5G Alliance, White Paper, Nov. 2018.

[4] I. Bashir, *Mastering Blockchain*, 2nd ed. Birmingham, U.K.: Packt, 2018.

[5] "Ericsson mobility report: Massive IoT in the city," Ericsson, Stockholm, Sweden, Rep. EAB-14, Nov. 2016.

[6] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst. (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[7] Z. Czirkos and G. Hosszú, "Peer-to-peer methods for operating system security," in *Encyclopedia of Networked and Virtual Organizations*. Hershey, PA, USA: IGI Glob., 2008, pp. 1185–1191.

[8] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. OSDI*, vol. 99, 1999, pp. 173–186.

[9] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *J. Ind. Inf. Integr.*, vol. 6, pp. 1–10, Jun. 2017.

[10] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: http://bitcoin.org/bitcoin.pdf

[11] *Bitcoin Developer Guide-Find Detailed Information About the Bitcoin Protocol and Related Specifications*. Accessed: Dec. 15, 2018. [Online]. Available: https://bitcoin.org

[12] C. Lee. *Litecoin*. Accessed: Apr. 4, 2019. [Online]. Available: https://litecoin.org

[13] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Wakefield, MA, USA, Ethereum Project, Yellow Paper, vol. 151, pp. 1–32, 2014.

[14] *Blockchain Finality in IoT: The Importance of Finality in Cross-Chain Communication*, Coinmonks, Bengaluru, India, Jul. 2018.

[15] S. King and S. Nadal, "PPCoin: Peer-to-peer crypto-currency with proof-of-stake," May 2019.

[16] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *arXiv preprint arXiv:1710.09437*, 2017.

[17] B. Xu, D. Luthra, Z. Cole, and N. Blakely. (2019). *EOS: An Architectural, Performance, and Economic Analysis*. [Online]. Available: https://www.whiteblock.io/library/eos-test-report.pdf

[18] "A distributed network for the smart economy," Shanghai, China, NEO, White Paper. Accessed: Nov. 25, 2018. [Online]. Available: http://docs.neo.org/en-us/whitepaper.html

[19] IoTeX Team. *IoTeX: A Decentralized Network for Internet of Things (IoT)*. Accessed: Jul. 12, 2018. [Online]. Available: https://iotex.io/white-paper

[20] R. Investment. *A Blockchain Platform for the Smart Economy—White Paper*. Accessed: Apr. 4, 2019. [Online]. Available: *https://heta.org/docs*

[21] *Watson Internet of Things-IBM Watson IoT Platform*, IBM, Armonk, NY, USA. Accessed: Nov. 25, 2018. [Online]. Available: https://www.ibm.com/internet-of-things/solutions/iot-platform/watson-iot-platform
[22] G. Greenspan. (2015). *IoT Chain: A High-Security Lite IoT OS—White Paper*. [Online]. Available: *http://www.multichain.com/download/MultiChain-White-Paper.pdf*
[23] D. Larimer, "Delegated proof-of-stake (DPoS)," Blacksburg, VA, USA, Bitshare, White Paper, 2014.
[24] A. Bessani, J. Sousa, and E. E. P. Alchieri, "State machine replication for the masses with BFT-SMART," in *Proc. 44th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw.*, 2013, pp. 1–8.
[25] R. Han, V. Gramoli, and X. Xu, "Evaluating blockchains for IoT," in *Proc. IEEE 9th IFIP Int. Conf. New Technol. Mobility Security (NTMS)*, 2018, pp. 1–5.
[26] N. Teslya and I. Ryabchikov, "Blockchain platforms overview for industrial IoT purposes," in *Proc. IEEE 22nd Conf. Open Innov. Assoc. (FRUCT)*, 2018, pp. 250–256.
[27] D. Schwartz, N. Youngs, and A. Britto, "The ripple protocol consensus algorithm," San Francisco, CA, USA, Ripple Labs Inc., White Paper, vol. 5, 2014.
[28] *Blockchain Speeds and the Scalability Debate*. Accessed: Feb. 28, 2018. [Online]. Available: https://blocksplain.com
[29] T. Stack. *Internet of Things (IoT) Data Continues to Explode Exponentially. Who Is Using That Data and How?* Accessed: Feb. 5, 2018. [Online]. Available: https://blogs.cisco.com
[30] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication," in *Proc. Int. Workshop Open Problems Netw. Security*, 2015, pp. 112–125.
[31] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.*, 2002, pp. 251–260.
[32] I. Adan and J. Resing, *Queueing Theory*, Eindhoven Univ. Technol., Eindhoven, The Netherlands, p. 180, 2010.
[33] M. Alaslani and B. Shihada, "Analyzing latency and dropping in today's Internet of multimedia things," in *Proc. IEEE Consum. Commun. Netw. Conf. (CCNC)*, 2019, pp. 1–4.
[34] S. Mubeen, S. A. Asadollah, A. V. Papadopoulos, M. Ashjaei, H. Pei-Breivold, and M. Behnam, "Management of service level agreements for cloud services in IoT: A systematic mapping study," *IEEE Access*, vol. 6, pp. 30184–30207, 2018.
[35] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, "Efficient Byzantine fault-tolerance," *IEEE Trans. Comput.*, vol. 62, no. 1, pp. 16–30, Jan. 2013.
[36] "Design best practices for latency optimization financial services technical decision maker," San Jose, CA, USA, Cisco Syst., White Paper, 2007.
[37] The Chippewa County Economic Development Corporation. *Building Guide for Manufacturers*. Accessed: Nov. 25, 2018. [Online]. Available: https://chippewa-wi.com
[38] Wikipedia. (2018). *List of United States Cities by Area—Wikipedia, the Free Encyclopedia*. Accessed: Dec. 13, 2018. [Online]. Available: https://en.wikipedia.org/wiki/List_of_United_States_cities_by_area
[39] K. I. Park, *Fundamentals of Probability and Stochastic Processes With Applications to Communications*. Cham, Switzerland: Springer, 2018.
[40] *Statista—Size of the Bitcoin Blockchain From 2010 to 2019*. Accessed: Apr. 21, 2019. [Online]. Available: https://www.statista.com/

Maha Alaslani received the B.Sc. degree from King Abdulaziz University (KAU), Jeddah, Saudi Arabia, in 2011, and the M.Sc. degree in wireless communication and green networks from the King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia, in 2015, where she is currently pursuing the Ph.D. degree in computer, electrical, and mathematical sciences and engineering.

She is a member of the Networking Research Laboratory, KAUST, supervised by Prof. B. Shihada. Prior to KAUST, she was a Teaching Assistant with the Information Technology Department, KAU. Her current research interests include Internet of Things (IoT), IoT security, and emerging technologies, such as software-defined networks and blockchain.

Faisal Nawab is an Assistant Professor with the Computer Science and Engineering Department, University of California at Santa Cruz, Santa Cruz, CA, USA, with a focus on intersection of distributed cloud computing and big data management. His current research interests include global-scale, edge-aware data management systems, and the transformation of cloud computing to a more distributed computing paradigm that utilizes resources around the world, closer to users.

Basem Shihada (SM'12) received the Ph.D. degree in computer science from the University of Waterloo, Waterloo, ON, Canada.

He is an Associate and a Founding Professor with the Computer, Electrical and Mathematical Sciences Engineering Division, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia. In 2009, he was appointed as a Visiting Faculty Member with the Department of Computer Science, Stanford University, Stanford, CA, USA. His current research interests include energy and resource allocation in wired and wireless networks, software-defined networking, Internet of Things, data networks, smart systems, network security, and cloud/fog computing.