Mitigating Messaging and Processing Load in IoT Environments Managed by Blockchain

Johannes Mittendorfer
Johannes Kepler University Linz
johannes@johannes-mittendorfer.com

Karin Anna Hummel
Johannes Kepler University Linz
karin_anna.hummel@jku.at

ABSTRACT

We study the use of Blockchain in an Internet of Things (IoT) environment where microservices are provided by sensors. Blockchain assures secure and trustworthy transactions, yet it causes high load caused by block creation on the numerous IoT nodes. To mitigate the load, sensor nodes are clustered and block creation is restricted to IoT cluster managers in the distributed Blockchain ledger. The concept is exemplified by a use case describing a neighborhood of smart homes that are equipped with environmental sensors such as dust, temperature, and precipitation sensors. We provide an implementation of the Blockchain IoT system comprising two consensus algorithms, Proof of Work (PoW) and Proof of Stake (PoS). By means of simulations conducted in Mininet, we derive first results which reveal that – depending on the sensor polling period – the service response time can be reduced by clustering. Further, clustering reduces PoW conflicts to about 30% due to creating fewer blocks.

CCS CONCEPTS

• Computing methodologies → Simulation evaluation; **• Computer systems organization → Sensor networks**; **• Software and its engineering → Distributed systems organizing principles**.

KEYWORDS

Blockchain, IoT, smart home sensors, simulation, Mininet

ACM Reference Format:
Johannes Mittendorfer and Karin Anna Hummel. 2019. Mitigating Messaging and Processing Load in IoT Environments Managed by Blockchain. In *17th International Conference on Advances in Mobile Computing and Multimedia (MoMM '19), December 2–4, 2019, Munich, Germany*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3365921.3365949

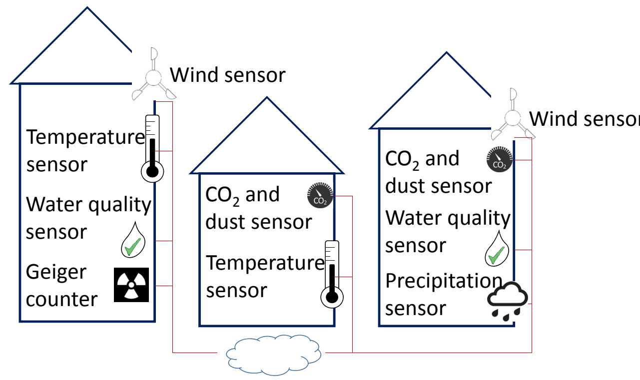

Figure 1: Schematic view of typical environmental sensors employed in a neighborhood of smart homes.

1 INTRODUCTION

Blockchain has become popular through Bitcoin, a system for maintaining a virtual currency based on distributed databases and transactions. As Blockchain refers to a general concept, it has been discussed beyond the financial sector as a promising paradigm for the energy market, smart cities and transportation, etc.

Due to the distributed nature of the Internet of Things (IoT), consisting typically of a many of networked sensors, tagged things, and machines, Blockchain is a technology of interest for managing IoT systems. In particular cooperations of multiple IoT providers can be well supported by Blockchain. Yet, the particularities of the Internet of Things, namely, low-latency communication, the large number of networked nodes with low processing power, and transmission of small data packets, require studies on whether and how Blockchain can be leveraged best.

In this work, we study the use of Blockchain in an IoT environment inspired by a *smart neighborhood scenario* as depicted in Figure 1. The scenario consists of smart homes with IoT devices which provide microservices in terms of simple environmental sensor measurements or aggregations. We assume that the IoT devices are owned by different parties. The system is not centrally managed and is open for new participants. As a consequence, the system should self-organize concerning the advertisement of services, trustworthy selection of services, and secure payment.

In this setting, our contributions are:

- We discuss how Blockchain can be employed in IoT environments (see Section 2 and Section 3). To mitigate inherent scaling problems of Blockchain, we employ an extension based on clustering. We implement the flat and the clustered Blockchain topology together with the major consensus algorithms Proof of Work (PoW) and Proof of Stake (PoS) as detailed in Section 4.
- We conduct a performance study and derive first results. In particular, we compare the flat Blockchain topology with a clustered topology in terms of timeliness and severity of conflicts in Section 5.

MoMM '19, December 2–4, 2019, Munich, Germany

ACM ISBN 978-1-4503-7178-0/19/12.
https://doi.org/10.1145/3365921.3365949

2 RELATED WORK

It is widely agreed that Blockchain technology offers benefits in IoT scenarios, in particular when the IoT sensors are shared between different parties [12] and microservices are provided by one or multiple sensors [11]. First, Blockchain provides a decentralized infrastructure to coordinate the nodes in the IoT network. Second, it provides a secure and trustworthy way to implement transactions [10]. Related work has both investigated how IoT devices can be connected and managed using Blockchain technology and how light-weight security mechanisms can be introduced in this frame. In particular provenance can be provided by Blockchain by storing the observed conditions of goods in (food) supply chains [4].

Similar to our clustering approach, related work typically foresees separating the Blockchain ledger network and connections to IoT devices. In [9], a LoRaWAN network is employed to connect the IoT devices to a Blockchain Ethereum network through a private gateway using smart contracts. The use of smart contracts for microservices is described in detail in [11]. We follow this approach by introducing advertisements and light-weight service contracts.

While assuring trust is a necessity, processing a full Blockchain algorithm on IoT devices which typically have limited processing power and storage capabilities is challenging. In [3], Twitter is leveraged and transaction security is based on chains of tweets in this novel Blockchain implementation. An approach most related to ours but focused on security and privacy is described in [5], where a clustered approach based on PoW is introduced for smart homes. We extend this work by an investigation of Blockchain conflicts and by including both PoW and PoS in our study.

3 EMPLOYING BLOCKCHAIN IN IOT

The Internet of Things typically refers to a network of tagged objects, autonomous devices, or sensors connected through Internet technologies. Application fields of IoT are manifold, such as, transportation, health-care, smart homes, and smart offices [1]. Connecting a large number of heterogeneous things efficiently requires plug'n'play mechanisms to scale. Thus, the IoT nodes have to support service advertisement and discovery, service use, and payment mechanisms. Finally, trust in services and transactions is needed [7].

To assure trust in services of a decentralized IoT, Blockchain may be adopted. The basic idea relates to an older concept, which describes procedures to ensure the correctness of a timestamp value cryptographically [6]. By applying encryption and using generated hash values from previous documents to create a new hash value, the order of events can be ensured. This procedure results in a chain of document signatures ordered by the timestamps of creation. It is not possible to maliciously introduce information with a fake timestamp between two blocks of the chain at a later point in time. Typically, Blockchain systems are distributed, meaning the blocks are generated at different sites, replicated, and a consensus algorithm provides means to assure that the chain remains valid although concurrent block creation takes place. In case blocks cannot be added to the main chain, they are kept for later use and are considered conflicts. The two mostly used consensus algorithms are Proof of Work (PoW) and Proof of Stake (PoS).

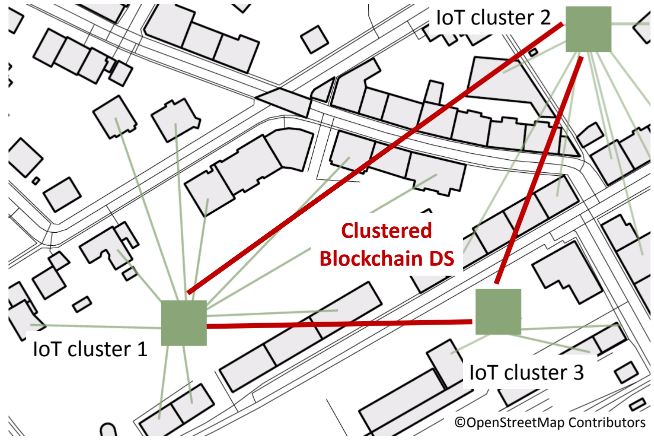

Figure 2: Formation of IoT clusters managed by co-located cluster mangers that represent the respective cluster in the Blockchain distributed system (DS).

3.1 Consensus Algorithms

PoW algorithms are based on time-consuming calculations necessary to create a block which are used to slow-down the block creation process. In the form used in this work, the calculated SHA-256 hash value used to sign a block has to comprise a given number of leading zeros (number of bits with a value of 0), expressed by the *difficulty* value. The higher the difficulty, the longer it takes to generate the hash value and thus the block. The difficulty increases with the number of blocks created, i.e., every 2016 blocks, the number of leading zeros is incremented by one. The PoW algorithm is used by the Bitcoin system [8].

PoS algorithms are based on the possession of a valuable that is traded in the network, such as coins, money, or goods. Participants who own a great amount of valuables are allowed to create more blocks than participants owning less. The time required to execute the calculations required for PoW can be reduced to a limited waiting time that a device has to wait before a block can be created. The more valuables a node owns, the shorter is the waiting time [2]. In our work, we use *points* to trade services and express the wealth of a node.

3.2 Blockchain System Architecture

In a fully connected network, all IoT nodes create blocks and participate in the consensus algorithms. This approach is obviously challenged by the scale of IoT environments and by the limited processing power and storage capabilities of IoT nodes. We will refer to this architecture as *flat* Blockchain distributed system (DS).

One option to mitigate block creation and messaging load is to introduce a hierarchy by grouping IoT nodes into clusters and to form a *clustered* Blockchain DS as visualized in Figure 2 for a smart neighborhood. The IoT nodes (typically sensors) send data to a cluster manager (typically a co-located switch, router, or gateway) which is the representative of the whole cluster and participates in the Blockchain system.

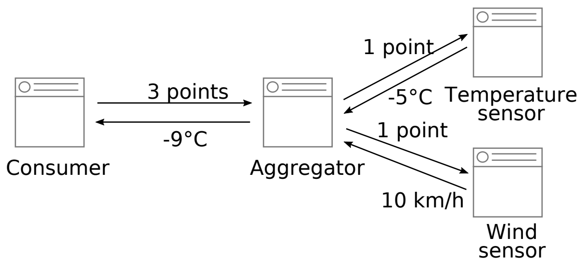

Figure 3: Windchill aggregator.

4 SYSTEM IMPLEMENTATION

The developed Blockchain client software is written in Python v3.6. Easy handling of data structures such as lists makes Python suitable for the implementation of a Blockchain system. In addition, the implementation can be well integrated into real IoT hardware and a network simulator, which we use to study the behavior of the IoT system managed by Blockchain. On all machines, the software modules of the client application are executed in a Docker container to isolate the software and to increase compatibility.

4.1 Block Format

The created blocks consist of six fields and are represented in JSON format: The *index* field refers to the position of the block in the chain and is incremented sequentially for each block added. The *previous hash* field contains the hash value of the preceding block. One field contains a *timestamp* which refers to the creation time of the block (synchronized clocks are assumed). The *nonce* field value is a random value used by the PoW algorithm. The *transaction* field consists of information concerning the trading of the service including, e.g., the amount of points transferred. Finally, the *hash* field contains an SHA-256 hash over all fields mentioned before.

For transmission, the JSON blocks are serialized. To request blocks of a node, lists of known neighbors, and debug information, standard HTTP GET requests are used. Through HTTP POST, a node notifies all other connected Blockchain nodes about a newly created block.

4.2 Software Architecture

The application separates the Blockchain specific and IoT specific software modules. The Blockchain modules implement the creation of blocks and chains, as well as the consensus algorithms. The IoT modules implement transmissions to other IoT nodes. In the flat Blockchain DS configuration, blocks are created by all IoT nodes and distributed among all nodes in the IoT system. Differently, in the clustered Blockchain DS configuration, Blockchain transmissions take place between the cluster managers only. In addition, cluster managers receive sensor information by querying the sensors in the respective cluster. Service tables are used to store services provided by IoT nodes, which further store context information (e.g., longitude and latitude). Several other queues are used to store pending service-related tasks such as transactions.

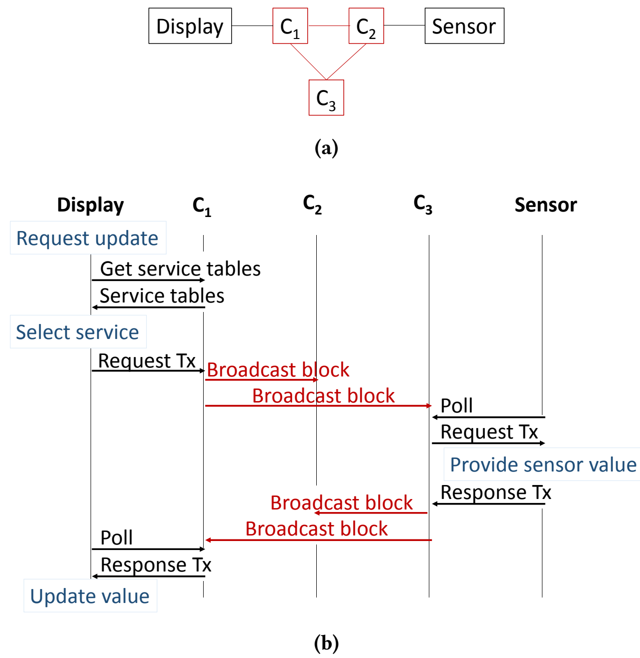

Figure 4: Message flow: (a) Example topology showing three cluster managers (C_1, C_2, C_3), one consumer (Display), and one service provider (Sensor); (b) Sequence of transactions (Tx) and messages triggered by a request of the Display.

4.3 Service Provisioning

A *service provider* offers a service such as a sensor reading that can be used by other devices. The *aggregator* fuses and aggregates data retrieved from a service provider. Finally, a *consumer* uses a service. Possible consumers are displays or database servers. Multiple roles may be taken by one IoT node.

To get a sensor or aggregated measurement, a transaction is started. The provider of the sensor service evaluates whether the consumer (or aggregator) has enough points to pay for the service and sends the value accordingly. Figure 3 shows an example of a windchill aggregator which needs measurements from a temperature sensor and a wind sensor to calculate the windchill factor for a consumer.

In the clustered approach, cluster managers are added. Assume an example topology consisting of three cluster managers (C_1, C_2, and C_3), one consumer (Display) and one service provider (Sensor) as depicted in Figure 4a. Figure 4b visualizes the process of requesting a service by the Display, resulting in blocks created and broadcast by the responsible cluster manager C_1. In the following, C_3 queries the Sensor and broadcasts the newly created block containing the sensor value. The Display retrieves the value from C_1. Note that also the cluster manager C_2 receives all block information.

Services are advertised in terms of provided value and price. In case multiple nodes offer the same service, the best service is chosen. How to define the best service is a matter of configuration depending on the concrete use case. For reasons of simplicity and as we do not focus on optimizing service discovery and selection, the

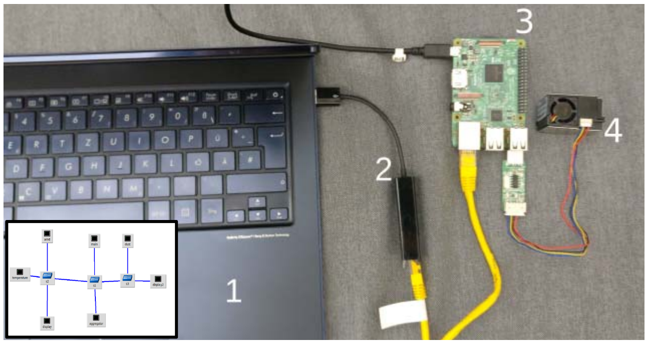

Figure 5: HW-setup external sensor: (1) Notebook running Mininet (with network topology snapshot), (2) Ethernet port configured in Mininet that connects the external device, (3) Raspberry-Pi 3 running service provider software, and (4) an USB/UART PM10/PM2.5 real dust sensor.

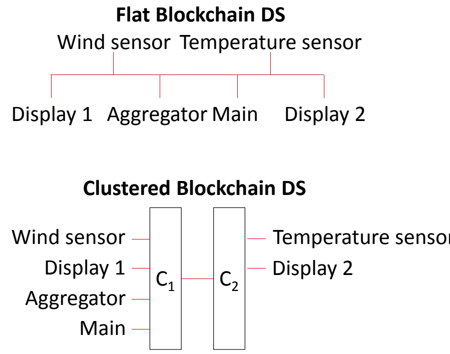

Figure 6: Flat and clustered topology of the Blockchain IoT distributed system used in the experiment (the Main node is the registrar of the network that maintains a database of available IoT devices).

current implementation uses a product to count in all parameters in order to select the best service s as follows

$$s = \operatorname*{argmax}_{x \in S} \frac{1}{d/D \times p/P \times 1/r \times t/T},$$

where S is the set of available services, $d, p, r, t > 0$. The parameters are normalized by the maximum value of each parameter. The product decreases with geographical *distance* d (D is the maximum distance) and *service price* p (P is the maximum price). Furthermore the product increases with the system parameter *reliability* r (fraction of requests that were answered in the past by this node) and decreases with the system parameter average *response time* t (T is the maximum response time).

4.4 Integration in Mininet

For studying the concept under varying network and system settings, we integrate the implementation in Mininet[1]. Mininet is a network simulator that allows to employ real code (kernel, switch, application code) to create a realistic virtual network. Later, the code can be deployed to real hardware.

Mininet is leveraged to create hosts, switches, and links by specifying the most relevant parameters of the link, while Ethernet, IP, and TCP implementations are already available to support the HTTP-based transmission of blocks. We further extend the virtual network by connecting real physical devices to even more increase realism. In the current implementation, we successfully integrated a Raspberry Pi computer with an attached dust sensor. This device is treated similar to the virtual IoT devices simulated in Mininet. Figure 5 shows the hardware setup needed to integrate an external sensor (here, a dust sensor) to Mininet.

5 EXPERIMENTS AND RESULTS

We run experiments, first, to investigate the benefits of the clustering approach in IoT settings. Second, we aim to dissect differences in using PoW and PoS. We study the following characteristics of the system:

- **Timeliness:** Timeliness is expressed as the *response time*, measured as the accumulated time it takes to create a block, transmit it to the destination, validate the block, create and transmit the response, and validate the response.
- **Severity of conflicts:** The severity is measured as the *number of conflicting blocks*, i.e., blocks that cannot be appended to the main chain; we further use the *ratio of conflicting blocks* which is the number of conflicting blocks related to the number of created blocks.

5.1 Experiment Setup

To compare the clustered with the flat distributed system architecture, we use the experiment topologies depicted in Figure 6. Both topology variants are implemented in Mininet. The used network settings are based on Wi-Fi, all links are modeled with a network delay of 5 ms, a bandwidth of 10 Mbit/s, and a loss rate of 0.02.

During one experiment, the Display device requests a new sensor value every 20 seconds, for a time of 15 minutes. The frequency has been selected to avoid interference with previous requests. Each IoT node (flat topology) or cluster manager (clustered topology) accumulates 5, 500 bytes necessary to store the chain of blocks in one experiment. Each experiment is run 20 times.

The consensus algorithms introduced in Section 3 are used. PoW is configured by its difficulty, i.e., 32 bits (32 leading zeros) are used at the start of the simulation. PoS is configured by a default waiting time of two seconds. In our example, points used by PoS are equally distributed between the cluster managers.

5.2 Timeliness

Figures 7 and 8 depict the response times achieved by the two different topologies employing PoS or PoW. The figures show that the median response times of PoW (6.21 s) and PoS (6.46 s) are similar, though the variance is much smaller for PoS. When using the clustering approach, the response time depends on the polling period used. We find that with a polling period of 5 s the median response time can be reduced to 5.03 s for PoW. Similar results are observed for PoS, not shown here.

[1]http://mininet.org/

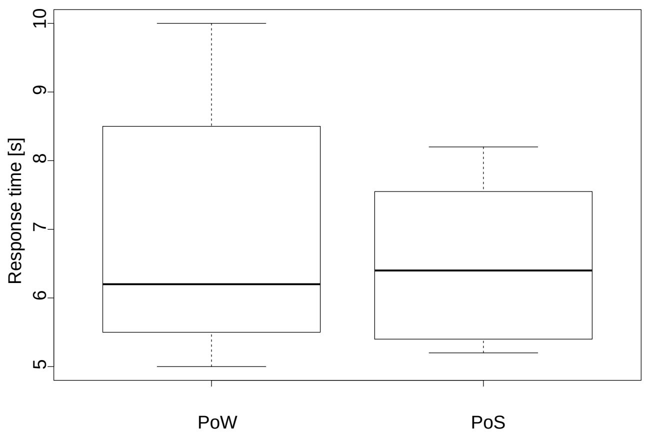

Figure 7: Flat Blockchain DS (PoW and PoS): Distribution of response time visualized as box-plots (thick middle line represents the median, box boundaries the 25th and 75th percentiles).

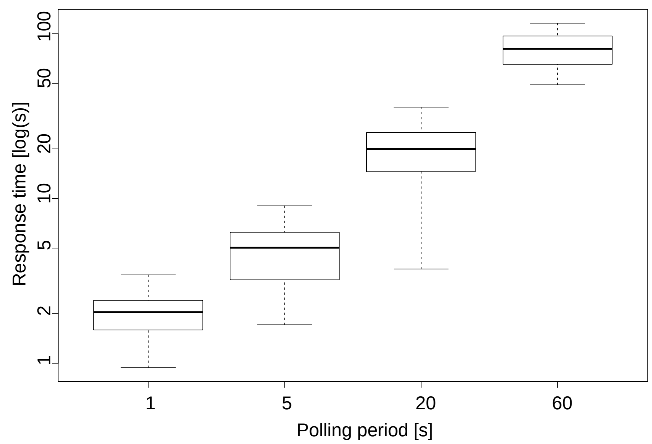

Figure 8: Clustered Blockchain DS (PoW): Distribution of response time with varying polling intervals visualized as box-plots (thick middle line represents the median, box boundaries the 25th and 75th percentiles).

5.3 Severity of Conflicts

Table 1 summarizes the results concerning conflicting blocks. Clustering can in particular reduce the number of conflicts for PoW to about 30% of the conflicts occurring in the fully connected network. With this improvement, PoW is feasible as long as the number of clusters does not grow drastically. PoS cannot benefit from clustering with respect to the number of conflicting blocks.

Table 1: Number of conflicting blocks and fraction of conflicting blocks.

	Flat		Clustered	
	Number	**Ratio (%)**	**Number**	**Ratio (%)**
PoW				
median	9.5	1.05	3	0.33
mean	10.8	2.29	3.27	0.36
std.	6.28	0.69	2.41	0.26
PoS				
mean	15	1.66	15	1.66
median	14.37	1.59	14.62	1.62
std.	7.46	0.82	7.55	0.83

6 CONCLUSIONS

We implemented a Blockchain system dedicated to an IoT environment that provides microservices. To reduce the number of conflicts caused by concurrently creating blocks, IoT devices are clustered and only cluster managers are allowed to fully participate in the Blockchain system. First results derived by Mininet simulations show that clustering can in particular reduce the number of conflicts for PoW to about 30% of the conflicts occurring in the fully connected network and, thus, enables the use of PoW in IoT settings. PoS is a more optimized consensus strategy and cannot reduce the number of conflicts by clustering. Clustering may also reduce the service response time when compared to the fully connected network, yet the improvement depends on the selection of the sensor polling frequency and comes with additional messaging necessary to request sensor data by the cluster managers.

REFERENCES

[1] Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The Internet of Things: A Survey. *Computer Networks* 54, 15 (2010), 2787 – 2805.

[2] Iddo Bentov, Ariel Gabizon, and Alex Mizrahi. 2016. Cryptocurrencies Without Proof of Work. In *Financial Cryptography and Data Security*, Jeremy Clark, Sarah Meiklejohn, Peter Y.A. Ryan, Dan Wallach, Michael Brenner, and Kurt Rohloff (Eds.). Springer Berlin Heidelberg, 142–157.

[3] Francesco Buccafurri, Gianluca Lax, Serena Nicolazzo, and Antonino Nocera. 2017. Overcoming Limits of Blockchain for IoT Applications. In *12th Int. Conf. on Availability, Reliability and Security (ARES '17)*. ACM, Article 26, 26:1–26:6 pages.

[4] Thomas K. Dasaklis, Fran Casino, and Constantinos Patsakis. 2019. Defining Granularity Levels for Supply Chain Traceability Based on IoT and Blockchain. In *Int. Conf. on Omni-Layer Intelligent Systems (COINS '19)*. ACM, 184–190.

[5] Ali Dorri, Salil S. Kanhere, Raja Jurdak, and Praveen Gauravaram. 2017. Blockchain for IoT Security and Privacy: The Case Study of a Smart Home. In *2017 IEEE Int. Conf. on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 618–623.

[6] Stuart Haber and W. Scott Stornetta. 1991. How to Time-stamp a Digital Document. *Journal of Cryptology* 3, 2 (01 Jan 1991), 99–111.

[7] Shancang Li, Li Da Xu, and Shanshan Zhao. 2015. The Internet of Things: A Survey. *Information Systems Frontiers* 17, 2 (01 Apr 2015), 243–259.

[8] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.com/bitcoin.pdf (online, accessed Oct. 25, 2019).

[9] Kazim Rifat Oezyilmaz and Arda Yurdakul. 2017. Integrating Low-power IoT Devices to a Blockchain-based Infrastructure: Work-in-progress. In *Thirteenth ACM Int. Conf. on Embedded Software 2017 Companion (EMSOFT '17)*. ACM, Article 13, 13:1–13:2 pages.

[10] Jianli Pan and Zhicheng Yang. 2018. Cybersecurity Challenges and Opportunities in the New "Edge Computing + IoT" World. In *2018 ACM Int. Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Sec'18)*. ACM, 29–32.

[11] Amir Taherkordi and Peter Herrmann. 2018. Pervasive Smart Contracts for Blockchains in IoT Systems. In *2018 Int. Conf. on Blockchain Technology and Application (ICBTA 2018)*. ACM, 6–11.

[12] Lei Xu, Nolan Shah, Lin Chen, Nour Diallo, Zhimin Gao, Yang Lu, and Weidong Shi. 2017. Enabling the Sharing Economy: Privacy Respecting Contract Based on Public Blockchain. In *ACM Workshop on Blockchain, Cryptocurrencies and Contracts (BCC '17)*. ACM, 15–21.