This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Novo Diaz, Oscar

Scalable Access Management in IoT using Blockchain : a Performance Evaluation

Published in:
IEEE Internet of Things Journal

DOI:
[10.1109/JIOT.2018.2879679](https://doi.org/10.1109/JIOT.2018.2879679)

Published: 01/06/2019

Document Version
Peer reviewed version

Please cite the original version:
Novo Diaz, O. (2019). Scalable Access Management in IoT using Blockchain : a Performance Evaluation. *IEEE Internet of Things Journal*, 6(3), 4694-4701. <https://doi.org/10.1109/JIOT.2018.2879679>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Scalable Access Management in IoT using Blockchain: a Performance Evaluation

Oscar Novo

Abstract—The rise of the Internet of Things (IoT) implies new technical challenges such as managing a universally vast number of IoT devices. Despite the fact that there are already a variety of secure management frameworks for IoT, they are based on centralized models, which limits their applicability in scenarios with a large number of IoT devices. In order to overcome those limitations, we have developed a distributed IoT management system based on blockchain.

In this paper, we compare the performance of our solution with the existing access management solutions in IoT. We study the delays and the throughput rate associated with the systems and analyze different configurations of our solution to maximize its scalability. The objective of the paper is to find out whether our solution can scale as well as the existing management systems in IoT.

Index Terms—Internet of Things, Access Management, Blockchain, LwM2M.

I. INTRODUCTION

The Internet of Things (IoT) refers to the connection of devices with constrained capabilities to the Internet. One of the main aspects of IoT is to facilitate sharing the resources of the constrained devices with other entities. To do so, the IoT devices have to be capable to manage who can access their resources (e.g. device A gives access to its temperature resource to device B) and how (e.g. user demand, technical restrictions to grant permissions to many devices etc.). However, most IoT devices are too constrained to store and process the necessary information to manage the access of their resources on their own. This becomes even more critical in the dynamic IoT scenarios where new devices are constantly added and deleted from the system, forcing every IoT device to update its local policy accordingly. On the other hand, given the ever increasing array of IoT devices[1], there is a requirement for global, yet highly secure and reliable IoT access management systems. As a result, new solutions have been defined by different IoT standards to mitigate these limitations. Those solutions are built on constrained protocols, such as the Constrained Application Protocol (CoAP) [2]. They manage the IoT devices in a centralized fashion via a server. However, a common disadvantage of all the existing IoT commercial solutions is that they are susceptible to a problem, the single point of failure, preventing IoT systems to scale. Besides, centralized architectures are designed for systems where the managed devices are static and belong to the same management community during their lifetime.

Oscar Novo is with Ericsson Research and Aalto University, Finland, e-mail: oscar.novo@ericsson.com. "Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org."

In IoT, however, in some cases the assumptions of the centralized architectures are inaccurate [3]. Many IoT devices will be mobile like in the vehicle-to-vehicle scenarios for IoT [4]. They may be handled by multiple managers like in the supply chain scenarios [5], and they may belong to different management domains during their lifetime.

In order to provide a more flexible solution, we have previously developed a blockchain-based system for managing the access to the resources in the IoT devices [6]. In our system, the entities called *managers* are responsible for managing the access control permissions of a set of IoT devices. The policy is stored in the blockchain and the management interactions with the different IoT devices are performed through the blockchain. A blockchain is a decentralized digital ledger that does not require central authorities for management. Additionally, the blockchain technology is tamper-proof and secure.

The decentralized solution has several advantages [6]. Firstly, the architecture can be used in isolated managerial domains. Thereby, every managerial domain has the freedom to manage the IoT devices while the access control policies are still enforced by the rules in the blockchain. Secondly, our proposal can manage simultaneous managers, and they can access or modify the access control policies of an IoT device concurrently. Thirdly, constrained managers can easily administer multiple IoT devices due to the fact that the IoT devices do not access the management information directly from the managers. At the same time, constrained managers do not need to be constantly connected to the system, which saves its energy. Fourthly, the managers and IoT devices are interacting through the blockchain network enabling cross platform communication.

Even though blockchain technology is developing fast, the performance of storing information into a blockchain cannot be comparable with the performance in a centralized database due to high latency of the blockchain nodes confirming transactions. For this reason, this work focuses on the performance of the IoT networks to obtain access control information from the blockchain. As opposed to the transaction operations, obtaining blockchain information from our system does not need to be confirmed by the blockchain network.

The paper aims to evaluate and compare our decentralized architecture with the existing management solutions for IoT. Specifically, it evaluates the performance of the existing centralized architectures against our solution. In addition, it evaluates whether our architecture meets our design goal of scalability by measuring the system with different configurations.

The paper is structured as follows. Section II gives an

Fig. 1: Protocol stack of NETCONF, RESTCONF, CoMI and LwM2M

introduction to the access management technologies in IoT as well as to the blockchain technology. Section III shows the related work. In Section IV, an introduction to our new decentralized architecture is given. Section V compares our architecture with the existing access management technologies in IoT. Finally, Section VI concludes the paper.

II. BACKGROUND

This section introduces some of the concepts and technologies used in this paper. It first gives an overview of the state-of-the-art technologies related to access management in IoT by presenting the relevant standards and specifications and their available implementations, respectively. Then, it briefly describes the blockchain technology on which our proposal is based.

A. Access Management Technologies in IoT

There are mainly two standard-based approaches related to access management in IoT. The first approach is developed by the Internet Engineering Task Force (IETF) and is called CoAP Management Interface (CoMI) [7]. The other approach is called LWM2M [8] developed by Open Mobile Alliance (OMA). Accordingly, this section presents a brief description of the main characteristics of both approaches.

– *CoAP Management Interface.* CoAP Management Interface (CoMI)[7] is a CoAP-based management interface founded on NETCONF[9] and RESTCONF[10] as illustrated in Figure 1. CoMI extends both protocols to support the CoAP operations and CBOR data formats[11]. CoMI, as well as NETCONF and RESTCONF is a YANG-based protocol[12] that intends to work in a stateless client-server fashion and is adapted to low power, lossy-constrained networked devices.

Even though there is a wide range of NETCONF and RESTCONF implementations, CoMI is a very new protocol and, to the best of our knowledge, no implementation of it is currently available.

– *Lightweight Machine to Machine.* OMA Lightweight M2M (LwM2M)[8] is a management protocol for the management of constrained devices. LwM2M defines the application layer communication between a server and a client and defines four logical interfaces: *bootstrapping*, *client registration*, *device management and service enablement*, and *information reporting*.

- *Bootstrapping* allows a LwM2M Bootstrap Server to manage the access control and configuration of a device in order to register it to a LwM2M Server.

- *Client Registration* is used by LwM2M Clients to register and de-register with one or more LwM2M Servers.
- The *Device Management and Service Enablement* is used by the LwM2M Server to access the information from a registered LwM2M Client.
- *Information Reporting* allows a LwM2M Server to receive notifications when the information of a registered LwM2M Client changed.

The LwM2M protocol utilizes CoAP[2] as the underlying transfer protocol over UDP and SMS bearers as Figure 1 shows. In addition, it defines a resource model where each piece of information made available by the LwM2M Client is a resource. The resources are further logically organized into Objects.

There are a number of LwM2M implementations in use, but only *Wakaama* and *Leshan* are, according to our knowledge, widely used. *Wakaama*¹ is an OMA Lightweight M2M C implementation designed to be portable on POSIX-compliant systems. *Wakaama* belongs to the open source community through the Eclipse foundation. *Leshan*² is an open source OMA Lightweight M2M server and client in Java. It is based on the Californium³ CoAP implementation and it is also part of the Eclipse foundation community. The other implementations such as *coap-shepherd*⁴ (a JavaScript implementation of the protocol), LWMQN⁵ (a project to modify LwM2M to support MQTT as transport protocol) or MBed⁶ (a C++ implementation of the protocol) are not covered in this paper.

B. Blockchain

Before introducing our system architecture, we first provide an overview of the blockchain technology, which will serve later to understand in more details the benefits that this technology brings to our system.

Since the introduction of Bitcoin by Satoshi Nakamoto [13] in 2009, the blockchain has grown in popularity. In this manner, blockchain has been applied to areas other than cryptocurrencies since its potential goes beyond Bitcoin. On the other hand, its strengths make it an ideal component for IoT solutions: it is designed to impede central authorities dictate the rules of the system, its stored information is fully auditable as well as public to all the peers, the transactions are validated by the consensus of all the peers, and the technology is tamper-proof and cannot be manipulated by malicious actors.

The blockchain is a digital ledger of past transactions. A transaction is an exchange of information between different entities that is broadcasted to the network. The transactions are stored in blocks in chronological order, and every block contains a hash of the previous block creating a chain of blocks. The first block in the chain, called *genesis block*, is the only block that does not contain the hash of the previous block. That block is almost always hardcoded into the software.

Blocks not only store transactions but can also store other types of digital information, such as *smart contracts* [14]. A smart contract is a computerized transaction protocol intended

¹ <http://www.eclipse.org/wakaama/>

² <http://www.eclipse.org/leshan/>

³ <https://eclipse.org/californium/>

⁴ <https://github.com/petereb/coap-shepherd>

⁵ <http://lwmmqn.github.io/>

⁶ <https://www.mbed.com/en/platform/mbed-client/>

to digitally satisfy common contractual conditions between different parties. Smart contracts allow the performance of credible transactions without third parties. These transactions are auditable and irreversible.

At times blocks can be added to the blockchain at the same time, creating two separate branches of the blockchain. Those branches are called *forks*. However, those forks will not last, since the longest computational branch will always be chosen and the others discarded. The longest computational branch is calculated based on the combined difficulty of that chain, not the total number of blocks. The blocks in shorter chains are considered invalid chains and are often called *orphan* blocks.

As stated above, the blockchain does not rely on any centralized authority and assumes that the nodes in the network may be unreliable. For this reason, consensus — where the peers of the blockchain network have to mutually agree on the information that is added in the blockchain — is a fundamental part of the blockchain network. As a consequence, many different consensus algorithms have been designed for this purpose [15]: proof of work (PoW), proof of stake (PoS), proof of burn (PoB), and proof of capacity (PoC) are some of them. The most famous consensus algorithm is the proof of work (PoW) algorithm first introduced in Bitcoin. In this algorithm, the peers (called *miners*) compete to solve a difficult mathematical problem based in cryptographic hash algorithm. The solution proves that a miner did spend a large amount of time and resources solving the mathematical problem. When a block is solved, the transactions inside that block are considered confirmed.

Mining in the PoW is also the mechanism used to introduce new cryptocurrency into the system. Miners are paid for any valid blocks added into the blockchain. This serves both purposes of disseminating the currency among the peers as well as motivating miners to keep mining.

The proof of stake (PoS) is an alternative to PoW where only the peers holding assets in the system can participate in the consensus. In the proof of burn, on the other hand, miners send some cryptocurrency to a verifiable unspendable address to prove a set of transactions. The proof of capacity (PoC) is very similar to PoW except that instead of computation, storage is used.

III. RELATED WORK

This work aims to provide a performance evaluation between our proposed management system and the standard-based access control technologies in IoT. The work also provides an overview of the standard-based solutions related to access control in IoT. Similarly, other studies have been conducted in the field; reference [16] provides a literature review of the different access control technologies in IoT. The review identifies the current research and commercial access control solutions used in IoT. However, it overlooks the work done by the standardization organizations in access management for IoT. Reference [17] briefly describes some of the traditional architectures commonly employed in the IoT access control. However, it also omits to describe the standard-based solutions.

Fig. 2: Decentralized Management System

Furthermore, our paper describes a new access management architecture using blockchain. Although there is previous work describing new blockchain-based access control frameworks in the literature [18], [17], they differ from our solution in a few aspects. Firstly, our solution defines the policy rules in a single smart contract and uses a single blockchain. In contrast, [18] creates a different smart contract for every resource-requester pair. Secondly, [18] and [17] include the IoT devices in the blockchain. Thirdly, [17] does not use smart contracts.

IV. OVERVIEW OF THE DECENTRALIZED MANAGEMENT SYSTEM IN IoT

Previously we have developed an architecture for arbitrating roles and permissions in IoT [6]. The architecture was created with the following principles in mind: *Mobility*. IoT devices can freely move between different administrative domains and still be managed by the access control policies stored in the blockchain. *Lightweight*. Due to their restricted capabilities, the IoT devices adopt our solution per se, without the need of any modification. *Scalable*. The decentralized form of our solution facilitates the management of numerous IoT devices. *Concurrency*. IoT devices can be managed by multiple managers at the same time. *Accessibility*. Even though some constrained managers may be inaccessible by the use of sleeping patterns, our solution makes the access control rules available at any time. *Resilient*. The solution does not have a single point of failure.

Figure 2 presents an overview of the access management architecture and its components [6]. The role of a *manager* and *agent node* can be played by many different actors depending on the different use cases. For instance, a service provider could play the *agent* role and their customers could play the role of the manager managing the IoT devices that they owned. All the different entities belong to the blockchain technology except for the wireless sensor networks and the *Management Hub* nodes. In fact, all the wireless sensor networks are connected to the *Management Hub* nodes in order to interact with the blockchain. We will go through the different components of the architecture in more details in the sections below. A more detailed description of the architecture can be found in [6].

Name	Method	Description
Register Manager	Tx	Registers a Manager into the system.
Register Device	Tx	Registers a Device into the system.
Add Manager to Device	Tx	Registers a Device under a Manager's control.
Remove Manager to Device	Tx	De-registers a Device from a Manager's control.
Add Rule	Tx	Adds an access control rule under a device.
De-register Manager	Tx	De-registers a Manager from the system. The manager should not manage any device before being remove from the system.
De-register Device	Tx	De-registers a Device from the system.
Revoke Permission	Tx	Deletes a policy rule from the system.
Query Manager	Call	Checks if a node is a Manager.
Query Permission	Call	Checks if a Device can access the resources of another Device.

TABLE I: Operations of the Smart Contract

A. Wireless Sensor Networks

A wireless sensor network (WSN) is a network of devices that allows constrained connectivity in applications with limited power and computing resources. The IoT devices associated with the WSNs do not belong to the blockchain network due to their limited memory, computational power, and/or energy availability. Furthermore, the IoT devices in the WSNs communicate among them using CoAP[2] and are uniquely identified in the blockchain network through public keys. Hence, the communication between the devices is encrypted through DTLS [19].

Devices are implemented using the LibCoAP library⁷. LibCoAP is a C implementation of CoAP that also provides a security layer through DTLS.

B. Managers

A *manager* is an entity responsible for managing the access control permissions of a set of IoT devices. *Managers* do not belong to WSNs and are the owners of the devices in the majority of the cases. Their role is to interact with the blockchain network on behalf of the devices since IoT devices cannot interact with the smart contract on their own.

After registration of the IoT devices under the *manager's* control, the *manager* can define specific access control permissions for them. Moreover, all the registered IoT devices in the system have to belong to a particular registered manager.

C. Agent Node

The *Agent Node* is a specific blockchain node responsible for deploying the smart contract. Once the smart contract is deployed successfully into the blockchain network, the *Agent Node* shares the smart contract's address among the other entities of the architecture.

D. Smart Contract

The management system described in this paper is ruled by the operations defined in a single smart contract. All the

Fig. 3: Data Structure in the Smart Contract

operations allowed in the management system are defined in the smart contract. The profile of the management function set is shown in Table I.

As Table I shows, there are two types of operations in a smart contract. The operations that are triggered by blockchain transactions and the operations that query the blockchain. The former needs to be validated by the miners and, consequently, stored in the blockchain, whereas the latter does not. Furthermore, the latter does not incur any fee or significant delay.

The constrained device information, manager information, and the access control policy details are stored in two different data structures in the smart contract as shown in Fig. 3.

In addition to that, *managers* are the only entities with the ability to interact with the smart contract through *transactions* in order to register, de-register and define new policies in the system. However, *Management Hubs* are only able to query permissions from the blockchain using the *call* method.

E. Blockchain Network

The blockchain technology chosen for our proof-of-concept implementation was Ethereum [20]. Ethereum is a programmatic platform that includes a Turing complete scripting language called *Solidity*⁸ that is used for implementing smart contracts.

The *miners* in the network help keep the network secure and stable by approving transactions and storing copies of the blockchain. The information is fully decentralized and tamper-proof.

F. Management Hubs

Management Hubs are located at the edge of the WSNs. A *Management Hub* is a JavaScript interface that translates the information encoded in CoAP messages, by the IoT devices, into RPC messages. The RPC messages are understandable by the blockchain nodes. The *Management Hub* is connected directly to a blockchain node. The interface uses the *web3* JavaScript API to communicate with Ethereum nodes through RPC calls and a CoAP JavaScript library⁹ to connect with the IoT devices.

⁷ <https://libcoap.net/>

⁸ <https://solidity.readthedocs.io/> ⁹ <https://github.com/mcollina/node-coap>

Fig. 4: Scenarios Evaluated in Our Decentralized Management System

The *Management Hub* only implements the *Query Permission* operation from the smart contract. As a result, IoT devices will only be able to request access control information from the blockchain using the *Management Hub*. That is so because the *Query Permission* operation does not incur any fee nor do the *Management Hub* or the IoT devices need to spend any cryptocurrency to obtain that information. Furthermore, the operation is not stored in the blockchain and, therefore, it does not need to be validated by the miners, considerably increasing its response time.

Multiple sensor networks can be connected to a *Management Hub* node and multiple *Management Hub* nodes can be connected to the same blockchain node enabling network scalability.

In the event of failure or malfunction of a blockchain node, a *Management Hub* can automatically connect to a different node. In case of a *Management Hub* failure, the sensor networks can connect to other *Management Hubs* too.

V. PERFORMANCE COMPARISON

This section is divided in two parts. The first part describes the setup and tools used for evaluating the management systems. The second part of the section provides a comparison analysis of our proof-of-concept implementation against the existing standard-based access management systems in IoT.

A. Setup

All the experiments were done on an Ubuntu-16.04 desktop with Intel®Core™i7-950@3.07GHz, a 240GB solid-state drive, and 16GB of RAM. To measure our experiments, we use a benchmark tool called CoAPBench¹⁰ that uses Californium¹¹ as a CoAP implementation baseline. CoAPBench is, to the extent of our knowledge, the only benchmark tool available for the CoAP protocol. However, the tool was too limited for our testing purposes, and we extended it with new functionality. After our modifications, CoAPBench was able to send PUT, POST, and DELETE messages, allowed specifying a payload in the request messages and supported LwM2M registration operations.

For the emulation of the blockchain in our decentralized management system, we used Docker¹² and the image *ethereum/client-go* of the Ethereum protocol. The IoT devices in our implementation ran our modified version of the LibCoAP library on the same machine. To avoid bias, we repeated all the experiments five times.

B. Comparison with the State-of-the-Art

We compared our system to the state-of-the-art LwM2M servers, *Wakamaa* and *Leshan*. We ran CoAPBench as a LwM2M client towards *Leshan* and *Wakamaa*'s LwM2M servers and increased the number of virtual clients in CoAPBench from 1 to 10,000 to simulate different network sizes. After registration into the LwM2M server, every virtual client stressed the LwM2M server by continuously sending registration update requests during 30 seconds.

We carried out similar experiments in our decentralized prototype. However, the experiments in our prototype were limited to WSNs and *Management Hubs*, as the rest of operations of the smart contract are transaction operations and need to be validated by the miners. That validation incurs big delays, which are necessary in order for the blockchain network to validate the transactions properly. However, the operations performed by the *Management Hub* to fetch the access control information of a specific IoT device are executed immediately. For this reason, we limited the evaluation of our decentralized system to two scenarios: in the first scenario, we conducted experiments to simulate different network size in a WSN connected to a single *Management Hub* where the virtual clients stress the *Management Hub* querying for the permission of a specific resource. The second scenario simulates a diverse number of concurrent IoT devices accessing a resource of another IoT device. Moreover, in the second scenario, the IoT devices request the access control policy from the blockchain

¹⁰ <https://github.com/eclipse/californium.tools>

¹¹ <https://eclipse.org/californium/> ¹² <https://www.docker.com>

Fig. 6: Latency

through the *Management Hub* before providing the information to each device. Both scenarios run CoAPBench as a CoAP client increasing the number of virtual clients from a range of 1 to 10,000, and both scenarios use a single *Management Hub*. Figure 4 depicts both scenarios in the *Single Management Hub* box.

The goal was to gain an understanding of the scalability of the existing access management implementations as well as the performance of our system. Gaining such an understanding is important, since the scalability of traditional access management systems depends on a single server while our proof-of-concept decentralized management implementation can handle several *Management Hubs* at the same time and an unlimited number of nodes in the blockchain network.

1) *Throughput rate*: First, we evaluated the throughput of *Leshan*, *Wakamaa* and our implementation. As shown in Figure 5a, *Leshan* performs best and its peak performance is at 16,000 requests per second with 100 simultaneous clients. We used a logarithmic scale in this specific graph to better represent the information.

Conversely, *Wakaama* reached its best performance earlier than *Leshan* with around 12,000 requests per second with 5 concurrent clients. However, its performance was initially higher than *Leshan*'s from 1 to 50 concurrent clients, and it was able to keep its peak performance up to 100 concurrent clients. At that point, the performance of both servers fall significantly until converging towards 4,000 requests per second with 10,000 virtual clients. As a matter of fact, our OS ran out of native threads after running 11,000 concurrent clients

in *Leshan* and abruptly stopped running. *Leshan* consumed too many resources for high concurrent rates. Then again, *Wakaama* continued running even with 11,000 virtual clients (not shown in the line graph).

Our implementation exhibits in Figure 5b lower performance compared to centralized systems. The performance when the size of the WSN changed from 1 to 10,000 concurrent clients is relatively steady over 790 requests per second. The second scenario — where a diverse number of clients contiguously request the resource information of a particular IoT device — is consistently flat to 390 requests per second, despite the number of IoT clients requesting the information.

Figure 5c shows the number of timeout requests during the 30 seconds of execution of the tests. As is presented in the graph, the timeouts increased consistently for all the systems except *Leshan*. *Leshan* was able to serve all requests with marginal timeouts close to zero despite the fact that it abruptly stopped beyond the limit of 10,000 consecutively virtual clients due to the lack of memory.

Compared to the state-of-the-art solutions, the performance of our implementation is one order of magnitude lower. The main limiting factor is the latency generated by the RPC calls between the *Management Hub* and the blockchain node as we show in section V-C.

2) *Latency*: Our second experiment consists of evaluating the latency of the request operations. We evaluated all the systems and plotted the results into cumulative distribution functions (CDF) to have a better understanding of the distribution of latencies. Figure 6 shows the latencies of *Leshan* (6a),

Fig. 5: Throughput: Single Management Hub Scenarios

Wakamaa (6b) and our implementation (6c, 6d). Additionally, the median latency of all the systems is depicted in Figure 7.

The most significant point in Figure 6a is the dramatic increase of the latency in *Leshan*. While the latency is relatively small for up to 1,000 concurrent clients, it drastically increases for higher concurrencies to the point that the average latency for 10,000 concurrent clients is almost 3 seconds. The median latency graph shows clearly that the latency soared steadily.

It is interesting to note that, even though the results are average over 5 runs, certain data of the *cumulative distribution functions* in *Wakamaa* does not produce a normal distribution function. The reason is the overall low latency of the operations in *Wakamaa* and the high standard deviation values for values higher than 1,000 virtual clients. For instance, the standard deviation values obtained for 10,000 virtual clients were 20 ms in average.

It can be clearly seen that *Wakamaa* performs best in terms of latency and *Leshan* has the poorest performance among all the systems. *Wakamaa* impressively keeps a low latency for a high number of virtual clients. Regarding our implementation, the operations connecting directly with the *Management Hub* exhibit more than 50% better performance

Fig. 7: Median Latency

than the operations going through the CoAP Server. However, our system can achieve drastically lower latency values than *Leshan* for higher concurrency factors (up to 1,000). Although the overall performance of our system is significantly low in terms of latency when compared to the overall performance of *Wakamaa*, it still achieves acceptable quality levels.

In conclusion, our system achieves good response times and stable throughput for a high concurrency factor.

C. Scalability

As indicated by the results shown before, the performance of the traditional access management systems is better in terms of throughput than our system. However, it was not our goal to accomplish a throughput as high as in an optimized centralized system. Instead, we favour horizontal scalability where WSNs are connected to multiple *Management Hubs*.

This section evaluates the *Multiple Management Hubs* scenarios described in Figure 4. In the first scenario, we simultaneously connect two *Management Hubs* to the same blockchain node to evaluate their performance. The second scenario attaches two *Management Hubs* to two blockchain nodes separately. In both scenarios, the devices of the WSN connect equally to any of the *Management Hubs*.

Figure 8a shows the throughput of both *Management Hubs*. In this case, the values of the throughput of every *Management Hub* are half of the throughput of a single *Management Hub* connected to a single blockchain node (referred to Figure 5b). In this scenario, there is a drastic performance loss, and attaching several *Management Hubs* into the same blockchain node splits the absolute throughput of the blockchain node (around 900 requests per second) among all the *Hubs*.

On the other hand, when we connect two *Management Hubs* to different blockchain nodes, the throughput is equated to the throughput in Figure 5b. In other words, our system is designed to obtain the best throughput performance when we attach multiple *Management Hubs* to different blockchain nodes. If we extrapolate our results further, twenty *Management Hubs* distributed geographically among different WSNs would outperform *Leshan* and *Wakamaa*.

It must be borne in mind that the results in Figure 8 only show a maximum of 5,000 virtual clients for each *Management Hub* instead of the 10,000 virtual clients of the other graphs. The reason for that is the design and implementation of CoAPBench. CoAPBench is implemented in Java programming language; every virtual client runs in a different *Thread* and the OS of our testing machine ran out of native threads

Fig. 8: Throughput: Multiple Management Hubs

when we performed our scalability experiments with 10,000 virtual clients for every *Management Hub*. We decided not to use several physical machines because the network delays would influence negatively our results and we tried to reduce external factors to minimum in our experiments.

VI. CONCLUSION

This paper presents a proof-of-concept architecture that adopts the blockchain technology to implement an access management system for IoT where the credentials and permissions to access the different IoT resources are stored globally into the blockchain. One of the main differences of our solution with respect to the existing ones is its decentralized design where the policy of the whole system is stored in a blockchain.

Furthermore, the paper evaluates the state-of-the-art access management systems in IoT against our solution. The aim of this study is to gain an understanding of the performance and scalability impact of distributing the access control information towards their IoT devices in comparison with the current IoT access management systems. Based from the findings of the study, our solution does not achieve better performance than the optimized centralized IoT systems in the case of a single *Management Hub*. However, the evaluation shows that our implementation has significant scalable advantages over the traditional scenarios when we distribute the load among the blockchain network's nodes.

In summary, our solution is designed to favour horizontal scalability where WSNs are connected to multiple *Management Hubs*. The evaluation in this paper proves the effectiveness of our design in some particular IoT scenarios.

ACKNOWLEDGEMENTS

The author would like to thank Nicklas Beijar and Mario Di Francesco for constructive criticism of the paper.

REFERENCES

- [1] "Ericsson Mobility Report: On the pulse of the networked society," Ericsson, Tech. Rep., June 2018. [Online]. Available: <https://www.ericsson.com/mobility-report>
- [2] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," RFC 7252 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–112, Jun. 2014, updated by RFC 7959. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc7252.txt>
- [3] X. Sun and N. Ansari, "EdgeIoT: Mobile Edge Computing for the Internet of Things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, December 2016.
- [4] C. Wu, "Connected vehicles and Internet of Things," in *2017 2nd International Conference on Telecommunication and Networks (TEL-NET)*, Aug 2017, pp. 1–1.
- [5] K. Pearsall, "Manufacturing supply chain challenges - globalization and IOT," in *2016 6th Electronic System-Integration Technology Conference (ESTC)*, Sept 2016, pp. 1–5.
- [6] O. Novo, "Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, April 2018.
- [7] M. Veillette, P. V. der Stok, and A. P. A. Bierman, "CoAP Management Interface," Internet Engineering Task Force, Internet-Draft draft-ietf-core-comi-02, Dec. 2017, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/html/draft-ietf-core-comi-02>
- [8] O. M. Alliance, "OMA Lightweight M2M, Technical Specification," 2018. [Online]. Available: <http://openmobilealliance.org/iot/lightweight-m2m-lwm2m/>
- [9] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, "Network Configuration Protocol (NETCONF)," RFC 6241 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–113, Jun. 2011, updated by RFC 7803. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc6241.txt>
- [10] A. Bierman, M. Bjorklund, and K. Watsen, "RESTCONF Protocol," RFC 8040 (Proposed Standard), RFC Editor, Fremont, CA, USA, Jan. 2017. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc8040.txt>
- [11] C. Bormann and P. Hoffman, "Concise Binary Object Representation (CBOR)," RFC 7049 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–54, Oct. 2013. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc7049.txt>
- [12] M. Bjorklund, "The YANG 1.1 Data Modeling Language," RFC 7950 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–217, Aug. 2016. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc7950.txt>
- [13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," <http://bitcoin.org/bitcoin.pdf>, 2008.
- [14] M. Wohrer and U. Zdun, "Smart contracts: security patterns in the ethereum ecosystem and solidity," in *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, March 2018.
- [15] N. Chalaemwongwan and W. Kurutach, "State of the art and challenges facing consensus protocols on blockchain," in *2018 International Conference on Information Networking (ICOIN)*, Jan 2018, pp. 957–962.
- [16] A. Ouaddah, H. Mousannif, A. A. E. Kalam, and A. A. Ouahman, "Access control in the Internet of Things: Big challenges and new opportunities," *Computer Networks*, vol. 112, pp. 237–262, 2017. [Online]. Available: <http://dx.doi.org/10.1016/j.comnet.2016.11.007>
- [17] O. J. A. Pinno, A. R. A. Gregio, and L. C. E. D. Bona, "ControlChain: Blockchain as a Central Enabler for Access Control Authorizations in the IoT," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec 2017, pp. 1–6.
- [18] A. Ouaddah, A. A. E. Kalam, and A. A. Ouahman, "FairAccess: a new Blockchain-based access control framework for the Internet of Things," *Security and Communication Networks*, vol. 9, pp. 5943–5964, 2016.
- [19] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security Version 1.2," RFC 6347 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–32, Jan. 2012, updated by RFCs 7507, 7905. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc6347.txt>
- [20] G. Wood. (2013) Ethereum: A Secure Decentralised Generalised Transaction Ledger. [Online]. Available: <http://gavwood.com/paper.pdf>

Oscar Novo received his M. Sc. degrees in Telecommunication Software and Computer Science from *Aalto University* (Finland) and from *Universitat Politècnica de Catalunya* (Spain), respectively. He is currently working at Ericsson Research in Finland and pursuing his PhD degree in Computer Science at Aalto University. His research interests include Internet of Things, blockchain, and machine intelligence.