Received: 15 December 2020 | Revised: 14 February 2022 | Accepted: 21 February 2022

DOI: 10.1002/cpe.6934

RESEARCH ARTICLE

WILEY

Securing the access control policies to the Internet of Things resources through permissioned blockchain

Alessandra Rizzardi[1] | Sabrina Sicari[1] | Daniele Miorandi[2] | Alberto Coen-Porisini[1]

[1]Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, Varese, Italy

[2]U-Hopper, Trento, Italy

Correspondence
Sabrina Sicari, Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, via O. Rossi 9 - 21100, Varese, Italy.
Email: sabrina.sicari@uninsubria.it

Abstract

Security and privacy of information transmitted among the devices involved in an Internet of Things (IoT) network represent relevant issues in IoT contexts. Guaranteeing effective control and supervising access permissions to IoT applications is a complex task, mainly due to resources' heterogeneity and scalability requirements. The design and development of highly customizable access control policies, along with an efficient mechanism for ensuring that the rules applied by the IoT platform are not tampered with or violated, will undoubtedly have a significant impact on the diffusion of IoT-based solutions. In such a direction, the article proposes the integration of a permissioned blockchain within an honest-but-curious (i.e., not trusted) IoT distributed middleware layer, which aims to guarantee the correct management of access to resources by the interested parties. The result is a robust and lightweight system, able to manage the data produced by IoT devices, support relevant security features, such as integrity and confidentiality, and resist different kinds of attacks. The use of blockchain will ensure the tamper-resistance and synchronization of the distributed system, where various stakeholders own applications and IoT platforms. The methodology and the proposed architecture are validated employing a test-bed.

KEYWORDS

blockchain, fog computing, Internet of Things, privacy, security, sticky policy

1 | INTRODUCTION

Security and privacy, along with scalability and interoperability, are the main critical issues, which can potentially be an obstacle to the widespread adoption of the Internet of Things (IoT) paradigm. Proper solutions for security and privacy management can be modeled, exploiting various mechanisms,[1] such as access-control policies' enforcement, policy life cycle management, encryption techniques, and so on.[2] However, such approaches must deal with a huge amount of data producers and consumers, which exchange heterogeneous data among each other, sometimes across different IoT applications, possibly owned by various stakeholders. The involved parties could not trust each other. Seamlessly, the IoT platform itself, which is responsible for managing IoT resources, could not be considered trusted, as is usually done. Hence, the possibility of tampering or violation of the IoT platform itself should be considered in defining a robust IoT network infrastructure.

Given the emerging issues related to security, privacy, scalability, and interoperability, a first viable research direction to cope with them consists in the adoption of fog computing principles.[3] Fog computing aims to operate as an intermediate layer among data consumers and producers, moving most of the computing tasks, which are typically performed by resource-constrained IoT devices or by a cloud, towards the edge of the network, bringing the services' provision closer to where data are effectively acquired, elaborated, and shared.[4] At the same time, complex tasks do not burden

IoT devices. Beyond the advantages in terms of reduced network's latency, consumption, and service response times (thus improving scalability and service availability), also security and privacy benefits can be obtained, since the fog layer does not represent a single point of failure, as, for example, cloud storage, which must also be considered trusted.[5] In fact, following a fog computing fashion, security and privacy functionalities and policies should also follow a similar approach if the information and network resources are managed at the network's edges. Also, the IoT platform could be treated as an honest-but-curious party since it does not directly handle third parties' applications.

The scope of this article is the definition of a fog-based IoT network infrastructure, where involved parties can cooperate securely without trusting each other. Blockchain technology is integrated to guarantee that IoT network components enforce the correct policies before disclosing data. The blockchain approach has been chosen because it allows services to operate in a decentralized or peer-to-peer fashion, without the need for a central authority or other trusted intermediaries.[6] Moreover, the adoption of blockchain, instead, for example, of multiple non-colluding authorities, appears the better choice because fewer entities are involved in the network, which should, in turn, be considered trusted (e.g., trust authorities, certification authorities).

Summarizing, the contributions of the article are the followings:

- The integration of permissioned blockchain mechanisms into security and privacy-aware IoT middleware is proposed, following a fog computing perspective. In fact, blockchain management is delegated neither to a central authority nor to end-devices, but it is in charge of the IoT management layer.
- The goal of adopting the blockchain is twofold: on the one hand, it let the IoT platform not be trusted; on the other hand, it helps in protecting policies, which regulate the access to IoT resources against tampering and violation. In this way, a malicious component of the IoT platform is prevented from changing the established rules without the consensus of the other components belonging to the network.
- A test-bed contributes to the validation and evaluation of the proposed approach in order to demonstrate its feasibility and reliability in the real world.

The article is organized as follows. Section 2 investigates the available solutions adopting blockchain in the IoT; Section 3 outlines the background of the presented work, detailing the considered IoT platform and pointing out the current weaknesses. Section 4 proposes the integration of the IoT platform with the blockchain technology, clarifying how the emerged issues are overcome; in Section 5 concrete experiments are described in order to validate the proposed approach and evaluate its performance. Section 6 ends the article, giving some hints for future developments.

2 | RELATED WORKS

Many researchers are attracted by the adoption of blockchain technology within IoT applications; such an approach is a consequence of decentralization, which means that efficiency, security, and privacy are perceived as pillars for a wide growth and diffusion of IoT scenarios in people's everyday life. For such reasons, many studies towards integrating blockchain and IoT are related to smart home use cases or, more in general, to smart cities.

For example, the work described in Reference 7 starts from the premise that blockchain techniques are computationally expensive and usually involve high bandwidth overhead and delays, not suitable for constrained IoT devices. Hence, the authors propose a new secure, private, and lightweight architecture for IoT, which exploits blockchain functionalities, but eliminates the overhead. Such a goal is obtained with a hierarchical architect consisting of a set of smart homes (the domain application considered in the le paper), an overlay network, and cloud storage, coordinating data transactions with blockchain to provide privacy and security. The approach proposed in such a paper pole apart from the one presented in our work, since we pursued the fog computing principles, replacing the use of the cloud in favor of a totally decentralized system.

Also focused on the smart home case study, the works in Reference 8, and Reference 9 exploit blockchain functionalities to realize an architecture,[10] which can guarantee security and privacy in an IoT fashion. Such a solution is similar to the one presented in Reference 7, where the following network's components are considered: a smart home, cloud storage, and an overlay. Note that only qualitative analysis is proposed as performance evaluation, except,[10] where quantitative analysis is conducted via simulations.

The work presented in Reference 11 is oriented towards the smart city context; the authors point out the need to put in action the blockchain mechanisms among smart devices at the communication level, which can include Bluetooth, 6LoWPAN, WiFi, Ethernet, 3G, and 4G as communication protocols. Obviously, the integration of existing communication protocols with blockchain reveals to be a significant challenge, since requirements vary from application to application. A potential solution envisioned by authors can be the implementation of multiple blockchains with the help of a blockchain access layer, in order to provide application-specific functionalities.

In such a direction, a possible solution is the hybrid blockchain architecture targeted to the IoT, presented in Reference 12, where the authors propose to split the IoT devices into subgroups, to further form sub-blockchains and achieve a distributed consensus. Relevant dimensions, such

as scalability, security, decentralization, efficiency, and network bandwidth, have been considered as guidelines in the subgroups definition, in order to achieve an optimal blockchain implementation, exploiting both Proof of Work (PoW) and Byzantine Fault Tolerant (BFT) mechanisms, as demonstrated by the conducted simulations.

The works proposed in References 13 and 14 are, instead, focused on a fully distributed access control system for IoT applications, based on blockchain technology. More in detail, Reference 13 proposes to exclude resource-constrained IoT devices from the blockchain, as we also suppose in our proposed solution. However, the approach pursued in Reference 13 operates only with a single smart contract. In contrast, in our approach, we make use of sticky policies (as detailed in Section 3) to manage access permissions for resources' disclosure, which are both lightweight and highly customizable. The main contribution of Reference 14 is that its framework introduces new types of transactions, which are used to grant, get, delegate, and revoke access. But, by adopting sticky policies, it is easy to update the access permission, since access's grants are contained in the policy itself, which travels along with the related data (for further details, please refer to Section 3.3).

The authors of Reference 15 propose a blockchain-based access control framework, called *Policychain*, which is responsible for enforcing Attribute-based Access Control (ABAC) policies, while the responsibility of ABAC policy administration and decision-making are offloaded to blockchain nodes. The envisioned scheme translates ABAC policies into blockchain transactions, hence the authors extended the blockchain inherent scripting instructions to support attribute acquisition of ABAC entities. Such an approach is based on a different policy management strategy, with respect to the adoption of sticky policies; they both seem valuable solutions to secure information within a blockchain-IoT network.

Another solution, again targeted to policy definition, is that presented in Reference 16, where policies are based on smart contracts, thus putting in act a self-enforcing agreement, embedded in computer code, managed by a blockchain. Three different policies are proposed: hardware and device security policies, access and authentication policies, and application security for the IoT network.

Wider envisioned research works are the ones proposed in References 17-22, which are detailed in the following.

The review proposed in Reference 17 identifies 18 use cases related to the adoption of blockchain technology in the IoT. Starting from such use cases, the main goal of such a work is to understand whether the blockchain along with peer-to-peer approaches can be employed to foster a decentralized and private-by-design IoT. Some open issues have been pointed out, such as adaptability, integrity, and anonymity. To cope with such emerging open challenges, the authors envision a layered architecture, as proposed in our article.

Similarly, the work described in Reference 18 aims to examine advantages in the use of blockchain technology and smart contracts for the IoT, which can be summarized as follows: robustness, failure-tolerance, transparency, verifiability, auditability of network's activity. Note that smart contracts are conceived as self-executing scripts that reside on the blockchain, containing contractual clauses into the code itself, to self-enforce them and minimize the need for trusted intermediaries between transacting parties, as well as the occurrence of malicious or accidental exceptions. In our proposed work, no smart contract is needed because sticky policies should not be executed along with a transaction, but they should only be stored inside the blockchain, to keep track of the access permission to the resources managed by the IoT network. The main reason for such a choice is that sticky policies are usually conceived to be directly attached to data, not to the blockchain. Hence, we can suppose that data can be detached from the blockchain, maintaining, at the same time, the link with the corresponding sticky policy. Further details on such a point can be found in Section 4. It is worth remarking that the scope of the article is also focused on how guaranteeing the reliability of the IoT platform, thus ensuring the correct application of policies on data disclosure.

The survey presented in Reference 19 examines the state-of-the-art of blockchain technologies. It proposes relevant scenarios for the so-called BIoT (Blockchain Internet of Things) applications in fields like healthcare, logistics, smart cities, and energy management. Such scenarios must face specific technical requirements, which must be addressed in order to obtain efficient and secure systems, such as security and privacy issues, energy consumption, network throughput, latency, bandwidth consumption, and so on.

A similar approach can be found in Reference 20, where, beyond possible integration between IoT and blockchain, also available blockchain platforms are described. Such a discussion is very interesting since each of the detailed platforms has peculiarities, which help understand the importance of modules required in a system using blockchain. However, such platforms result too complex with respect to our proposed solution; hence we decided not to adopt them but to directly integrate the blockchain primitives in our system, as detailed in Section 4. The main reason behind such a choice is that the target of our solution is IoT-constrained environments, which naturally require lightweight mechanisms to run efficiently.

A complete vision about the integration of IoT and blockchain is provided in Reference 21: here, the focus is on architectural aspects and industrial applications. Instead, the work in Reference 22 presents an in-depth description of blockchain technologies, in terms of applicability to IoT scenarios.

3 | SYSTEM'S BACKGROUND AND MOTIVATIONS

This section presents the background related to the proposed solution. First of all, the section explains the adopted IoT middleware platform, along with its main features; then, the sticky policy approach, used for the definition of a proper enforcement framework is described,

pointing out the drawbacks of the current approach, to motivate the introduction of the blockchain technology as a solution to the emerged weaknesses.

3.1 | Networked smart object middleware platform

The authors in Reference 23 defined the architecture considered in this article, which represents a flexible and cross-domain middleware, named *Networked smart object (NOS)*, tailored to the IoT environment.

NOSs are able to manage, in a distributed manner, the data provided by heterogeneous sources and evaluate, by means of proper algorithms,[24] the security and data quality of the information, in order to allow the users to be aware of the levels of reliability and trustworthiness of the services gathered by NOSs themselves. NOSs also provide a lightweight, and secure information exchange process, based on an authenticated publish&subscribe mechanism,[25] adopting the MQTT protocol. Finally, an enforcement framework, which is integrated within NOSs, monitors the correct application of the established sticky policies.[23]

Note that the need for middleware is motivated by the fact that IoT networks usually generate a huge amount of information, which cannot be efficiently handled in a centralized way (i.e., employing a single server or cloud); the introduction of an intermediate "edge" layer aims to process data closer to the sources, exploiting distributed and more powerful gateways, and reducing the amount of traffic to be delivered to the central server/cloud part. Hence, the middleware improves the usage of resources, thus providing a way better support to applications/services with tight latency constraints.

The following sections will describe the architectural components of NOSs, along with some details about their design and implementation. Moreover, the weaknesses, mainly in terms of security, of the current platform will be clearly put in the light.

3.2 | NOS's components

Two main entities compose a typical IoT system: (i) the nodes, conceived as heterogeneous data sources (e.g., RFID, NFC, actuators, sensors, social networks, etc.) which generate data for the IoT platform; (ii) the users, who interact with the IoT system through services making use of such IoT-generated data, typically accessing them by means of a mobile device (e.g., smartphone, tablet) connected to the Internet (e.g., through WiFi, 3G, or Bluetooth technologies).

Therefore, proper interfaces for the communications of NOSs with the data sources (i.e., the nodes) and with the users have been defined. More in detail, protocols based on HTTP or CoAP are adopted to collect the data from the IoT devices and allow sources' registration. In fact, NOSs deal both with registered and non-registered sources. The registration is not mandatory, but it provides various advantages in terms of security, since registered sources may specify an encryption scheme for their interactions with NOSs, thus increasing the level of protection of their communications (encryption keys' distribution is made by the algorithms presented in Reference 26). The information related to the registered sources is put in the storage unit, named *Sources*. Instead, for each incoming data, both from registered and non-registered sources, the following information is gathered: (i) the kind of data source, which describes the type of node; (ii) the communication mode, that is, how the data are collected (e.g., discrete or streaming communication); (iii) the data schema, which represents the content type (e.g., number, text) and the format of the received data; (iv) the data itself; (v) the reception timestamp.

Since the received data are of different types and formats, NOSs initially put them in the *Raw Data*storage unit. In such a collection, data are periodically processed in batches by the *Normalization* and *Analysis* phases, in order to obtain a uniform representation and add useful metadata regarding the security (i.e., level of confidentiality, integrity, privacy, and robustness of the authentication mechanism) and data quality (i.e., level of accuracy, precision, timeliness, and completeness) assessment. Such an assessment is based on a set of rules stored in a proper format in another storage unit, named *Config*, and are detailed in Reference 24; it allows users who access the IoT data to directly filter by themselves the data processed by NOSs, according to their personal preferences.

Instead, Message Queue Telemetry Transport (MQTT) protocol* is used for disseminating the information to the interested users. To this end, a topic is assigned by NOSs to each processed data. In order to manage the access to resources based on the assigned topics and the active policies, the original MQTT protocol has been further extended with *AUPS (AUthenticated Publish&Subscribe system)*,[25] and integrated with the enforcement framework based on sticky policies.[23] Figure 1 summarizes the NOS's components just introduced.

NOSs modules interact among themselves through *RESTful* interfaces; this allows the NOSs' administrators to add new modules or modify the existing ones at runtime since they are able to work in parallel and non-blocking manner. Moreover, the non-relational nature of the adopted *MongoDB* database also allows the data model to evolve over time dynamically. NOS interfaces have been implemented in a real prototype, which is openly accessible under a permissive license Apache v.2 at https://bitbucket.org/alessandrarizzardi/nos.git. *Node.JS*† platform has been used for developing NOSs' core operations, *MongoDB*‡ has been adopted for the data management, and Mosquitto§ has been chosen for realizing the open-source MQTT broker. To know more details about the implementation, please refer to Reference 24.

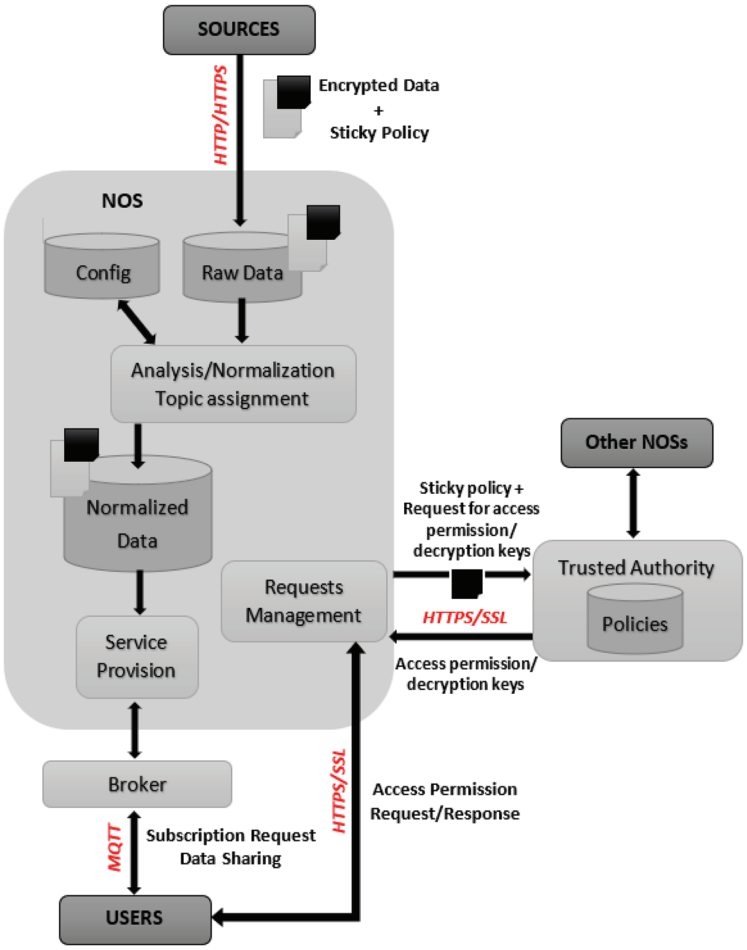

FIGURE 1 NOS architecture

3.3 | Sticky policies and drawbacks

NOSs are based upon the sticky policy paradigm, which enforces access control rules on producers' behalf. Note that multiple NOSs can naturally form a fog layer, which has direct contact with data sources and also with data consumers. Hence, a fully distributed and synchronized system is achieved.[27] The adoption of sticky policies is motivated by the fact that they are directly attached to the data. Consequently, they can flow with the data themselves across one or more application domains, being effectively enforced in a distributed manner. Sticky policies also enable access based on data owners' preferences and, thus, closer to the producers, with respect to an external central entity (e.g., a server or a cloud). Essentially, a sticky policy consists of an executable script containing the access permission to the resource contained in the transmitted information.[28] Its importance, in the IoT context, is related to the fact that different stakeholders may participate in the data provision and sharing of the resources; therefore, by directly attaching the policy to data, the co-existence of unrelated parties do not imply that the IoT platform manages the access control rules among heterogeneous information.

Karjoth et al.[29] first proposed a sticky policy paradigm, in order to provide a method for driving access control decisions and policy enforcement on a data-driven basis. In fact, sticky policies are transmitted along with data they refer to and "travel" with data across the entire system, thus protecting the whole data life cycle. Specifically, sticky policies usually include the following useful information:

$$sticky\ policy = \{data_{ow}, set_{sc}, data_{val}, set_{rule}\}, \tag{1}$$

where:

- $data_{ow}$ is the owner of data;
- set_{sc} represents one or more purposes for which data can be used;

- $data_{val}$ is a timestamp that points out the validity (i.e., the lifetime) of data within the IoT system; once this time has elapsed, then data must be discarded and no longer transmitted;
- set_{rule} includes one or more constraints that represent the rules to be applied to data (e.g., with whom the data may be shared if data must be shared in an aggregated form or not, and so on).

Hence, sticky policies allow specifying access rules in a highly fine-grained manner. Emerged features are particularly interesting in some scenarios, like those related to IoT, where users' or business' confidential information may flow across organizational boundaries,[28] but also when heterogeneous actors are involved within a highly distributed architecture.

Note that the sticky policy concept has already been integrated into the NOS platform, as presented in Reference 23. NOSs own no policies/credentials, because a *Trusted Authority* (TA) is responsible for: (i) the definition and management of a dictionary of valid scopes and constraints; (ii) providing NOSs (upon reception of a query) with access decisions related to the current sticky policy as well as decryption keys for accessing the data of interest. Moreover, no synchronization or policy sharing is required among multiple NOSs, since the TA manages access permissions. The behavior of the current system can be summarized as follows (see also Figure 1 for an overview of involved entities and interactions):

1. Data sources/IoT devices, owned by users, can agree with NOSs on an encryption scheme and on encryption keys to cipher the data transmitted, as specified in Section 3.2. Such a feature represents the first level of protection, basically covering the communication path. Further levels of security are provided by the enforcement mechanism realized by means of sticky policies. With data, the relevant policy is also sent, which is properly encrypted. The policy specifies how NOSs should manage the data;
2. The owner of data sends it in an encrypted way along with the associated sticky policy towards the nearest NOS;
3. When a NOS receives data from a source with the associated sticky policy, it first stores it in *Raw Data* storage unit; then, once data has been analyzed by NOS, following the procedure described in Section 3.2, a topic is assigned, and data is notified to interested subscribers;
4. Access to topics is regulated on-demand by NOSs through proper requests to TA, by means of the associated sticky policy; hence, NOS can contact the TA in order to obtain the access permissions for guaranteeing access to data.

Despite the resulting enforcement framework is highly customizable, it is not so efficient in large scale systems and imposes certain strict security requirements; in fact, the following weaknesses can be pointed out:

- Data sources/users must trust the IoT platform (composed by NOSs and TA) that their policies are correctly stored, applied, and transmitted along with the associated data under specific topics, during the time; in other words, a security mechanism, to protect the rules (i.e., the policies) themselves, still lacks;
- The presence of a single TA may present scalability issues as well as it could represent a single point of failure in the system.

Hence, the goal of our article is to overcome such issues, introducing blockchain technology, coupled with sticky policies, as explained in the next section. More in detail, the TA is removed from the system, and policy regulation is managed through a permissioned blockchain handled by the network of NOSs.

4 | INTEGRATION OF BLOCKCHAIN TECHNOLOGY IN NOS MIDDLEWARE PLATFORM

Blockchains have recently attracted the interest of stakeholders across different industrial activities, ranging from finance (the first and most famous use is in cryptocurrency), healthcare, product traceability, real estate, smart cities, smart homes, and so on,[30] thus paving the way for their adoption in the various application context and, more specifically, in IoT.[31] But, which are the real advantages of blockchain for IoT? They can be summarized as follows:

- Decentralization: there is no need for a centralized authority in charge of supervising the system's behavior and dictating rules or policies to be applied at each time; moreover, transactions are validated by all network's components (in our case, NOSs), thus avoiding to delegate such a task to a central entity;
- Distribution of information: since each network's components (in our case, NOSs) keep a copy of the blockchain, there is (again) no need for a centralized authority that keeps the information private;
- Data transparency and auditability: since a full copy of every transaction added to the system is stored in the blockchain and is public to all peers, then it is always possible to trace and monitor what happens in the network, guaranteeing that operations are legitimate;

- Robustness: blockchain is tamper-proof, so it cannot be manipulated by malicious parties (a discussion about attacks and vulnerabilities can be found in Section 4.2).

Hence, by adopting a blockchain, applications that usually run only through a trusted intermediary can now operate in a decentralized fashion, without the need for a central authority, and achieve the same functionalities. In order to understand how to pursue such an integration in the NOS middleware platform, some notions regarding the security mechanisms put in the act by blockchains must be clarified.

A block in a blockchain contains, in general, the following information:

$$block = \{id, set_{trans}, ts, hash_{curr}, hash_{pre}, sm\}, \tag{2}$$

where:

- *id* is the identifier or block number;
- set_{trans} is a set of transactions (i.e., the data content);
- The *ts* registers the timestamp when the block is created;
- $hash_{curr}$ is an authenticated data structure (e.g., a cryptographic hash) in charge of ensuring block integrity;
- $hash_{pre}$ is a reference to the hash of the preceding block, to identify the current block's place in the blockchain; note that, since each block references the hash of the block that came before it, a link between the blocks is established, thus creating a chain of blocks, named blockchain;
- *sm* (optional) are the so-called smart contracts, which are scripts that allow the coding and execution of computing programs on the blockchain itself; note that such scripts may contain access policies, which can be specified and enforced by the blockchain itself, preventing unauthorized operations on data.

Currently, NOSs must contact the TA to decide if a requesting consumer is entitled or not to access a certain kind of information (i.e., note that data sharing is managed based on a topic, as explained in Section 3.2. Such communications flow across an HTTP/SSL channel, as shown in Figure 1. But, what happens if a NOS becomes malicious and decides to follow no longer the rules imposed by the TA? In this case, IoT resources could be released to non-authorized parties or not legitimate operations could be done on data without the necessary permissions. To solve such an issue, the IoT platform must ensure to behave properly. However, data producers and consumers cannot be forced to trust NOSs (i.e., the IoT platform), and we must suppose that NOSs could be tampered with or violated by external parties. Also, in this situation, we must ensure that the IoT system continues to behave correctly, following the established policies. For such a reason, NOSs become responsible for managing the blockchain, which includes all the access permissions to be put in the act on the IoT resources. The role of the TA is no longer needed. In fact, the scope of the integration of the blockchain is that all NOSs follow the same rules; in this sense, the decentralized ledger guarantees the tamper-resistance and the synchronization of the operations, thus obtaining a consistent behavior of the IoT distributed system.

To summarize, NOSs, which manages both data and policies, should be considered trusted; otherwise, the integrated enforcement framework would be unreliable. In order to overcome such an issue, an integration of the existing NOS platform with blockchain technology is proposed hereby. The reason for choosing the blockchain approach is that it allows applications to operate in a decentralized or peer-to-peer fashion, without the need for a central authority or other trusted intermediaries.[6] Hence, NOSs should no longer be considered trusted; they will be regarded as honest-but-curious. Moreover, the adoption of blockchain, instead, for example, of multiple non-colluding authorities, appears the better choice, due to the fact that less entities are involved in the network, which should, in turn, be considered trusted. The kind of blockchain chosen for carrying out the integration is a permissioned (i.e., private or federated/consortium) blockchain, instead of a public (i.e., permissionless) one, due to the following motivations:

- The network of distributed NOSs will manage the blockchain, and no other participant is required; as a consequence, there is no reason to use a public blockchain; in fact, the middleware will be deployed for specific applications and domain context, which represent a sort of "close" system (e.g., a smart home, a traffic control system, a healthcare application, etc.).
- Both permissioned and permissionless blockchains are distributed ledgers; this means that there will be multiple versions of the same data stored in different places and connected through the NOSs' network, which is the basic principle of blockchain technology.
- Permissioned blockchains offer better performance than those of permissionless ones, because a limited number of nodes (i.e., the NOSs) participates in the blockchain itself; hence, they provide short latency for transactions' confirmation and for verifying the validity of a block.[30]
- Permissioned blockchains guarantee more efficient governance, since the administrators require less time to update the rules over NOSs' network.

Once clarified that there is no need to make public the blockchains managed by NOSs, it is fundamental to understand that, although we suppose that NOSs do not trust each other, each of them knows the entire set of its peer nodes participating in consensus for establishing the validity of a block. The method for achieving the consensus adopted in the proposed solution is the Byzantine Fault Tolerant (BFT) replication approach, since it better fits the requirements of the investigated scenario, such as a limited number of nodes (i.e., NOSs) involved in the blockchain management, low latency, and power consumption.[32]

More in detail, the security of blockchain protocols is obtained by means of different methodologies, which may be employed to achieve consensus on the newly added blocks. Robust consensus mechanisms are required to safeguard security for a self-regulating system such as blockchain, where there is no central authority, but a peer-to-peer network for validating transactions. The main exploited approaches to consensus are PoW[33] and BFT replication.[34] In this work, due to the nature of envisioned IoT scenarios, BFT replication is the best choice, thanks to its good performance (i.e., computational cost and latency) for a relatively small number of replicas; as just introduced in Section 1, only NOSs are involved in blockchain management, while data sources are excluded, in order to prevent the occurrence of scalability issues in the IoT network, which may potentially include a large number of data sources themselves. Note that PoW-based solutions sit the opposite end, since PoW-based blockchain offers good scalability with poor performance, due to the fact that the so-called mining task is particularly computationally intensive. Moreover, we have to consider that the mining of blocks is also time-consuming, while, in most IoT applications, low latency represents a fundamental requirement.

Summarizing, the proposed solution consists, on the one hand, in exploiting the secure blocks of blockchain to encapsulate the rules creation/update/revocation to protect them from tampering and violation by NOS's network (which is no longer considered trusted), and avoiding the presence of a central TA; on the other hand, to cope with computational and delay issues related to blockchain protocol, a fog layer of NOSs is added to the IoT network, in order to support IoT devices in performing the heaviest security tasks. Instead, in the previous version of NOSs' network, each NOS was independent of each others without any control over their behavior. Technical details about the presented approach are provided in the following.

4.1 | Technical details

The integration of blockchain within the IoT network, originally including NOSs and the sticky policy-based enforcement framework, required some changes, which are properly depicted in Figure 2, reporting the new version of the platform and its comparison with the one sketched in Figure 1. The main changes concern the disappearance of the TA and the introduction of the permissioned blockchain, which is managed only by the NOSs belonging to the IoT network.

More in detail, each NOSs can receive different kinds of data from a multitude of data sources/producers. As before, such information are analyzed, normalized, and assigned to proper topics, as described in Section 3.2. The broker notified users/consumers interested in the various topics when new data related to such topics were available. The end-users can decrypt the received information if the execution of the sticky policy allows them to access the ciphered data, as detailed in Section 3.3. Access permission and authorized operations on data are regulated by proper rules, as just explained. Such policies are the same for all NOSs belonging to the same IoT network. The scope of blockchain guarantees that one or more malicious NOS cannot self-manipulate the policies within the IoT system. In fact, all changes in policies (i.e., creation of a new policy, update of an existing one, or revocation) must receive the consensus of NOSs' network. In this way, the IoT system is able to work even in the presence of malicious parties. Moreover, the blockchain maintains a sort of log/monitoring of the system's rules. In fact, being the blockchain unchangeable, we automatically keep a trace of all changes happening within the IoT system. In this sense, new introduced sticky policies are stored as data in each created transaction.

Such behavior is obtained in the following way: (i) when an administrator of the IoT system requires a command of policy's creation/update/revocation, NOS creates a new block; (ii) NOS registers each new added block and other NOSs, belonging to the same IoT network, must reply on the consensus; (iii) NOS's database stores the blocks (i.e., *Policy* storage unit in Figure 2. It is worth remarking that, exploiting the fact that NOS has a modular architecture (as detailed in Section 3.2), the adding of new modules, responsible for the tasks related to blockchain technology, does not compromise the existing functionalities.

New modules mainly consist in the integration of *crypto-js* library ¶, which contains proper cryptographic functions, compliant with the Node.js based implementation of NOS, and in the definition of a new library, including novel functions for putting in act the blockchain mechanisms. With respect to the cryptographic functions, we choose *SHA256*, due to its wide use in *Bitcoin* context[35] and to its intrinsic robustness. Instead, regarding the blockchain library, the following main constructors and functions have been defined:

- *Transaction*: it is the constructor of object of type *transaction* (i.e., *trans*), characterized by an identifier *id*, a source (i.e., *src*, which is the NOS which initiates the transaction itself), a destination (i.e., *dest*, which is the network of NOSs belonging to the IoT system), a timestamp *ts*, and the policy *pl*, as the transaction's content;

$$trans = \{id, src, dest, ts, pl\}. \tag{3}$$

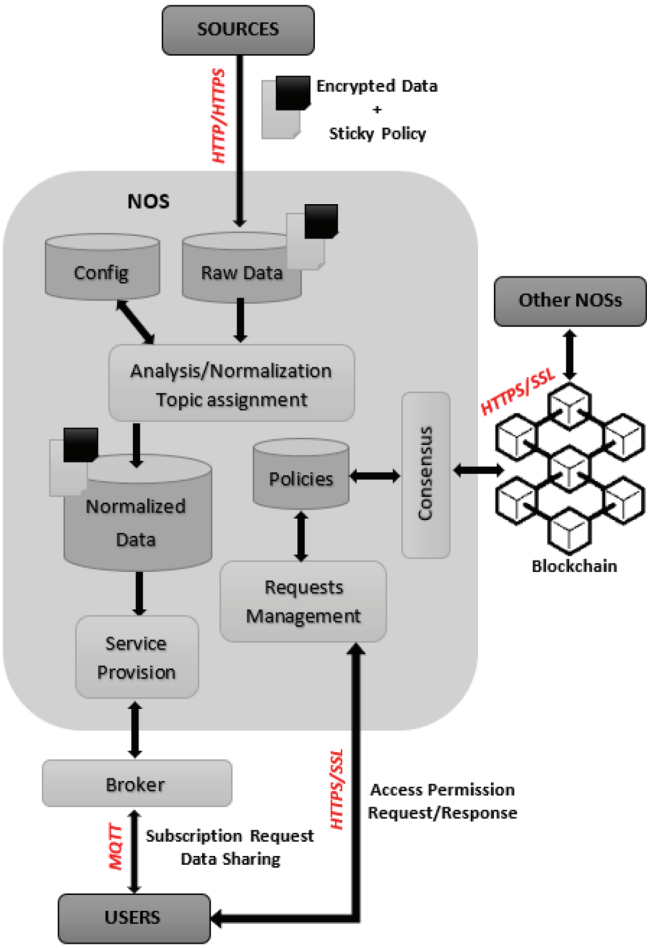

FIGURE 2 NOS architecture with the blockchain integration

- *Block*: it is the constructor of more complex object of type *block*, characterized by an identifier *id*, a set of transactions (i.e., set_{trans}), a timestamp *ts* (which may be different from that of the transactions), a cryptographic hash (i.e., $hash_{curr}$), calculated, as just said, by means of *SHA256* algorithm, the cryptographic hash of the previous block in the blockchain (i.e., $hash_{pre}$), and a *nonce*; note that the field $hash_{pre}$ is added to the block after the block is linked to blockchain (i.e., after verifying its validity-see *isChainValid* function);

$$block = \{id, set_{trans}, ts, hash_{curr}, hash_{pre}, nonce\}. \tag{4}$$

- *Blockchain*: it is the constructor of the object of type *blockchain*, which is responsible for the creation of the blockchain itself and for keeping track of pending transactions;

```
blockchain =
creation +
pending transactions' tracing
```

- *consensusBlock*: it is the function responsible for keeping track of the consensus related to a block; note that such a task is delegated to NOSs, which represents the more powerful entities in the proposed IoT scenario;

```
consensusBlock =
hash calculation + obtain consensus
```

- *isChainValid*: it is the function responsible for validating the transactions, until reaching the BFT consensus

```
isChainValid =
blocks' validation after consensus
```

It is worth remarking that the network of NOSs executes such operations in a parallel manner: Algorithm 1 summarizes the operations executed by a NOS; in fact, NOSs cooperate in maintaining valid and robust blockchain, while they acquire and process data coming from the IoT connected devices (Steps 1–3 in Figure 3). When a NOS receives a notification of policy creation, update, or revocation, it setups the corresponding transaction, by associating the information shown above (Steps 4 and 5 in Figure 3). Then, it notifies other NOSs of the newly generated block,

Algorithm 1. Policy creation, update, or revocation

Upon a policy *pl* creation/update/revocation
NOS calculates *trans*(*id*, *src*, *dest*, *ts*, *pl*)
block(*id*, *trans*, *ts*, $hash_{curr}$, $hash_{pre}$, *nonce*) generation
for all NOSs belonging to the IoT network **do**
    execute *consensusBlock*(*block*)
    compute *hash*(*block*)
    wait for reaching consensus
    isChainValid(*block*)
    if reached consensus **then**
        add block to blockchain
    else
        discard block
    end if
end for

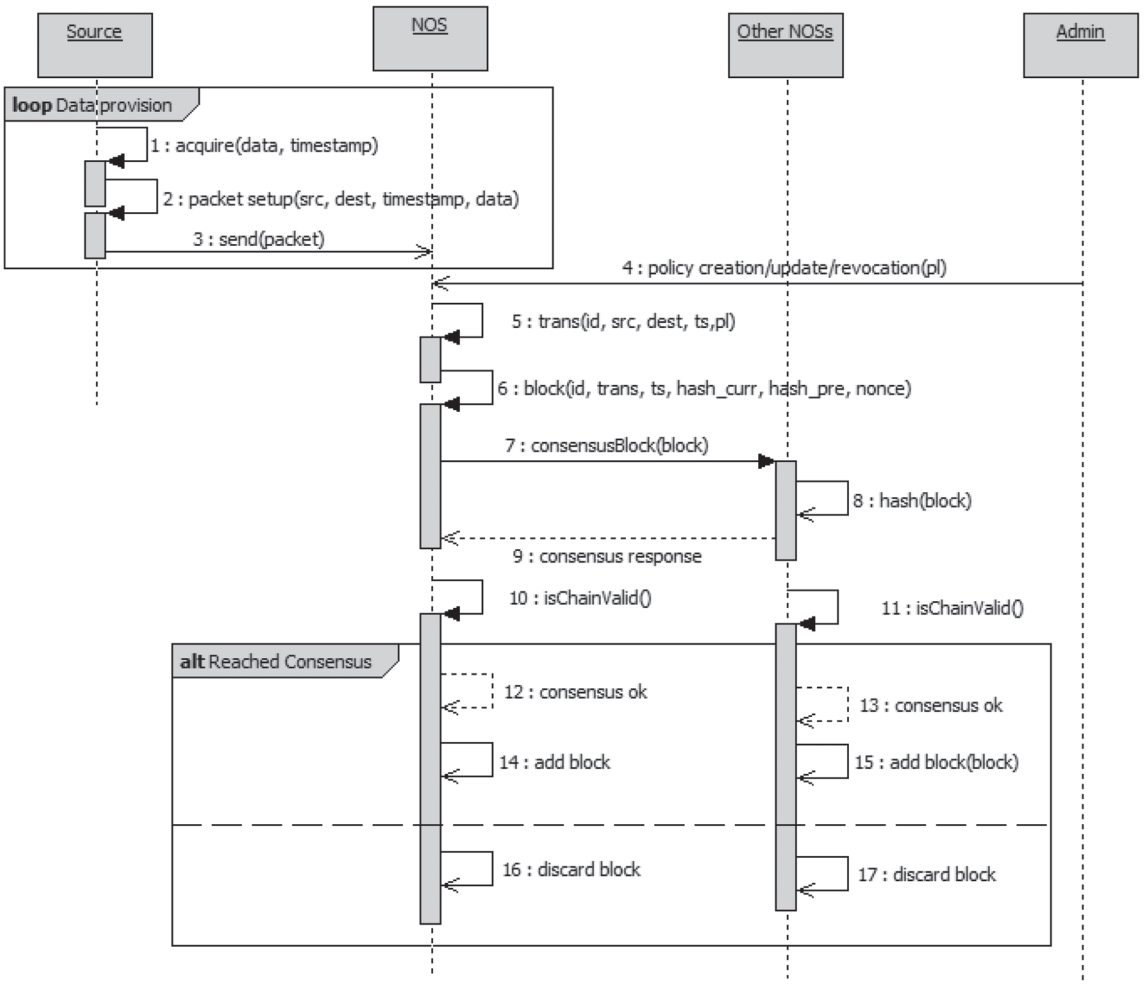

FIGURE 3 IoT system's interactions—data source and policy management perspective

requesting consensus for adding it to the blockchain (steps 6–11 in Figure 3). In detail, in order to reach the consensus, each NOS computes the hash. Note that the consensus is reached when the hash computed by each NOS is the same. Once reached the consensus, each NOS calculates the new hash value (i.e., $hash_{curr}$) to be associated to the new block, before adding it to the blockchain (Steps 12–15 in Figure 3), by executing the *isChainValid* function. Note that, if the consensus is reached, the block is valid, and the new defined/modified policy is added to the blockchain and becomes active in the IoT system; otherwise, the block must be discarded (Steps 16 and 17 in Figure 3).

4.2 | Security analysis

An IoT network can suffer from different kinds of attacks. In this section, we discuss how the NOS middleware platform can react or prevent possible threats, as follows:

- **Attacks towards data integrity and confidentiality**: data are secured by means of encryption techniques and by clever management of the access to resources by means of sticky policies. Moreover, NOSs are responsible for reliably handling the access permissions. Such an aspect is guaranteed by the adoption of the blockchain paradigm, because blockchain is inherently resistant to the modification of its content (since blocks, once added to the blockchain, are immutable);
- **Violation of the access control system**: the robustness of the access control is both ensured by the adoption of sticky policies and by the introduction of blockchain, for preventing misbehavior of some NOSs, which are not considered trusted;
- **Denial of Service**: NOSs are able to cope with such a kind of attack, thanks to a proper mechanism defined in Reference 36.
- **51% or majority attack**: security of the blockchain is granted if the majority of the network correctly validates the blockchain itself. However, if almost the 51% of the IoT network is compromised, then invalid blocks may be propagated, thus causing serious damages, such as the diffusion of false information. It is worth remarking that this is a very expensive attack, mainly in large-scale environments; hence, small networks are the most vulnerable. Since NOSs put in act proper mechanisms for continuously assessing the reputation of connected entities,[24] we expect that the system is able to recognize the majority attack at an early stage, thus promptly blocking it.

Concerning the last point, there is an important consideration to be made. In fact, we can suppose the existence of two different situations, in response to the presence of a malicious NOS:

1. If a NOS is tampered or violated, there is no way of blocking it; hence, it will continue to misbehave, but without affecting other NOSs (i.e., this is the advantage provided by the blockchain);
2. Each NOS is forced to follow the rules dictated by the administrator; hence, if a NOS is recognized as malicious, it can be blocked or restored. However, to actuate such a solution, a *Trusted Platform Module (TPM)* must be introduced. The *TPM* has the role of monitoring NOSs activity and recognizing any issues in their behavior.

5 | VALIDATION AND EXPERIMENTS

This section demonstrates the feasibility of the proposed solution through a real test-bed; four instances of NOS, running on four Raspberry Pi platforms, and a variable number of data sources, which virtually run on a personal computer, constitutes the test-bed. Such sources, for validation purposes, use data from real-world smart home test-bed[#]. In particular, we use data from smart meter number 2 of *Home A*, which include, among the others, electricity consumption data of kitchen lights, bedroom lights, duct heater HRV, and HRV furnace. Note that the home has eight rooms and includes three full-time occupants. Measures are acquired by means of installed sensors that collect electricity data every minute for the entire home. To obtain more details about the deployment and data, please refer to Reference 37. Wi-Fi connections are adopted for communications among the personal computer and the Raspberry Pi platforms (i.e., the NOSs). The same Wi-Fi connection is used for communications with the MQTT broker. Figure 4 sketches the simulation setup.

In order to demonstrate the effectiveness of the proposed solution, a comparison in terms of computing effort, storage overhead, and latency has been carried out. Table 1 summarizes the parameters setup in the test-bed. More in detail, NOS fetches data at 10 or 20 packets per second, corresponding to data acquisition time from sources. The data rate changes, in order to basically infer about the scalability of the system. The testing scenario has: (i) 14 data sources; (ii) the dimension of a block containing a single transaction is approximately equal to 580 bytes (calculated considering the blocks' and transactions' fields, shown in Section 4.1). Moreover, simulations have been measured over a period of 24 h.

In order to evaluate the efficiency of the proposed solution, a policy change is requested every 0.5 s; while, in reality, policy changes are not so frequent. Such a frequency, along with the number of connected data sources, obviously influences the memory occupancy as well as the computational effort. Note that the policy changes' frequency strictly depends on the application domain. Hence, it can range from a few times a year, as in a smart home scenario, where access control policies (e.g., remote monitoring of house's current state, access to the electrical data-set) could change when new people start or end to live in the house, or in case of absence for vacation, to few minutes or seconds in real-time systems, as in the case of hierarchical environments/companies (e.g., military applications), where, for example, sudden changes could happen and, as a consequence, the current policies could depend on the people currently present at a certain moment (e.g., higher authorizations could be disclosed to people owing a lower grade, for a limited period, in case of emergency).[38]

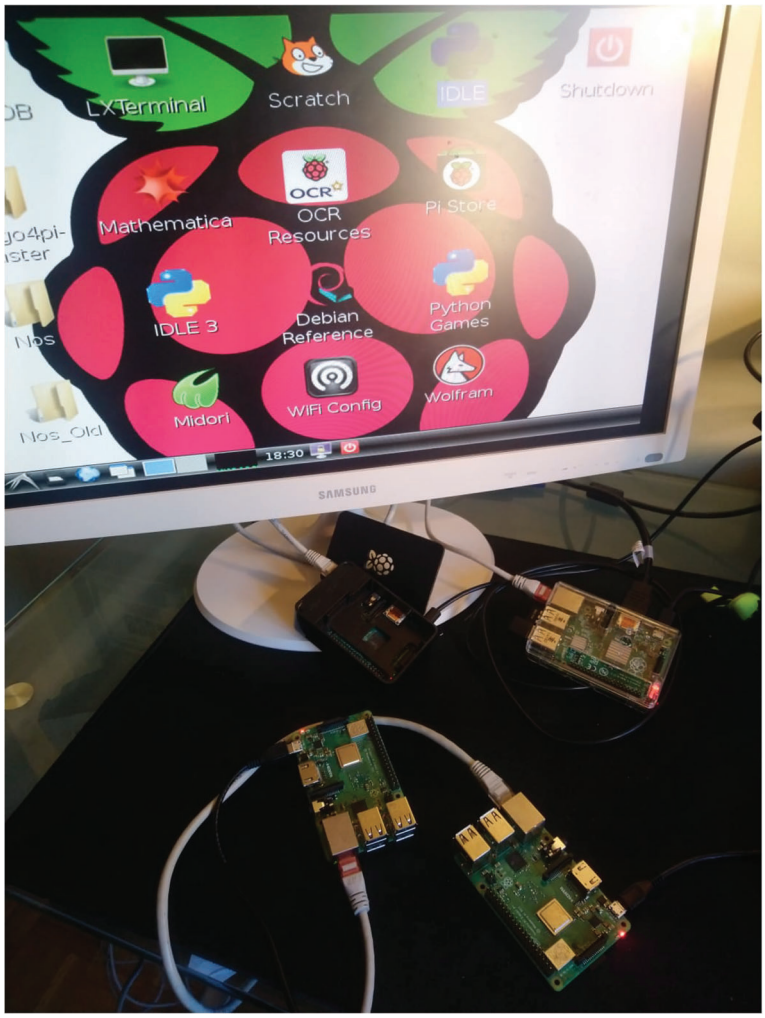

FIGURE 4 Test-bed

TABLE 1 Test-bed parameters

Parameter	Value
NOSs	4
Data sources	14
Data rate	10 and 20 packets/second
Policy change rate	0.5 requests/second
Block generation time	1 and 2 block/minute
Block dimension	580 byte
Observation time	24 hours

5.1 | Storage overhead

As regards storage capacity, the different components of the IoT system have the following storage requirements:

- Data sources have to store credentials for ciphering the data to be transmitted to NOS, as agreed during the registration phase (i.e., only in case of registered source). Then, they normally send the acquired data to NOS, along with the attached sticky policy. Such a kind of behavior does not differ from that performed in the NOSs' system version without the integration with blockchain. Hence, no storage or bandwidth overhead is recognized on IoT devices/producers. Hence, resource-constrained IoT devices do not have disadvantages from the adoption of the new solution;

- NOSs do not support the persistent storage of IoT data and related sticky policies for *Raw Data*and *Normalized Data* collections. Still, they have to maintain the blockchain's structure. Note that the size of the blockchain increases with each transaction. Since NOSs run on Raspberry Pi platforms, the maximum storage capacity with the actual technology corresponds to 1 gigabyte (i.e., the RAM provided by Raspberry Pi 2 and 3). With the integration with blockchain, the memory occupancy at runtime increased up to 835 KB on average compared to the previous version. Note that such results, shown in Figures 5 and 6, do not depend on the frequency of data fetching from sources, but only from the frequency of policies'

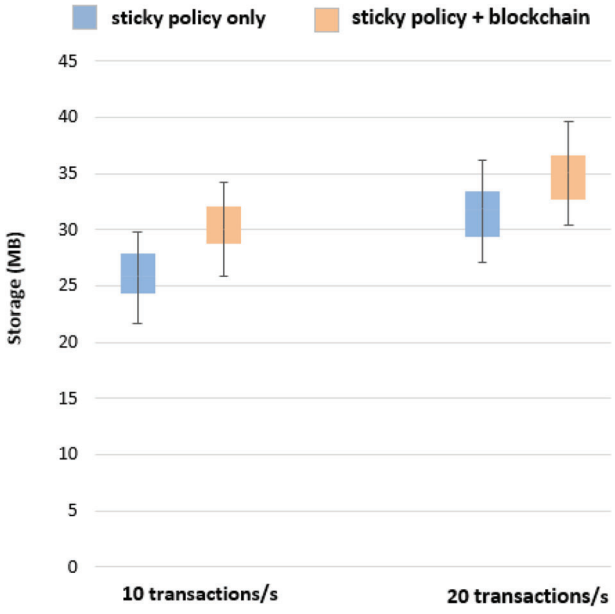

FIGURE 5 Average storage occupancy on NOSs: Whiskers-box diagram with and without blockchain

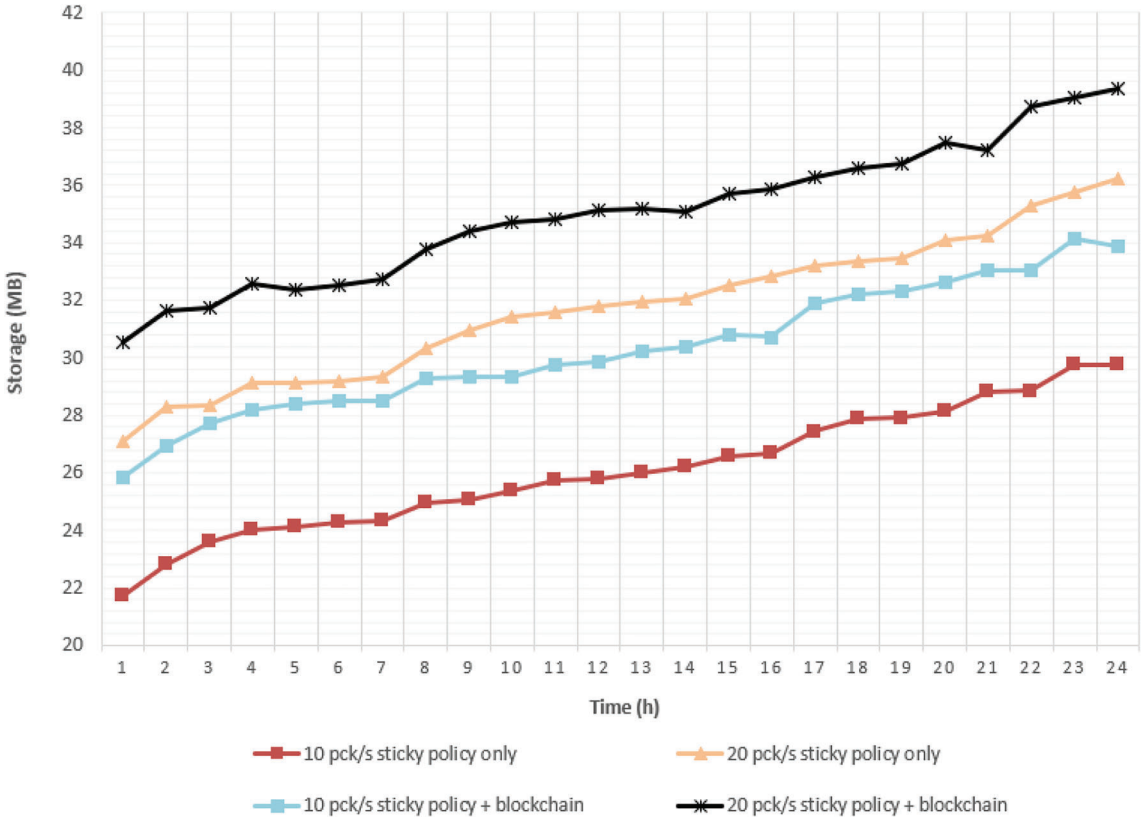

FIGURE 6 Average storage occupancy on NOSs: Line graph with and without blockchain

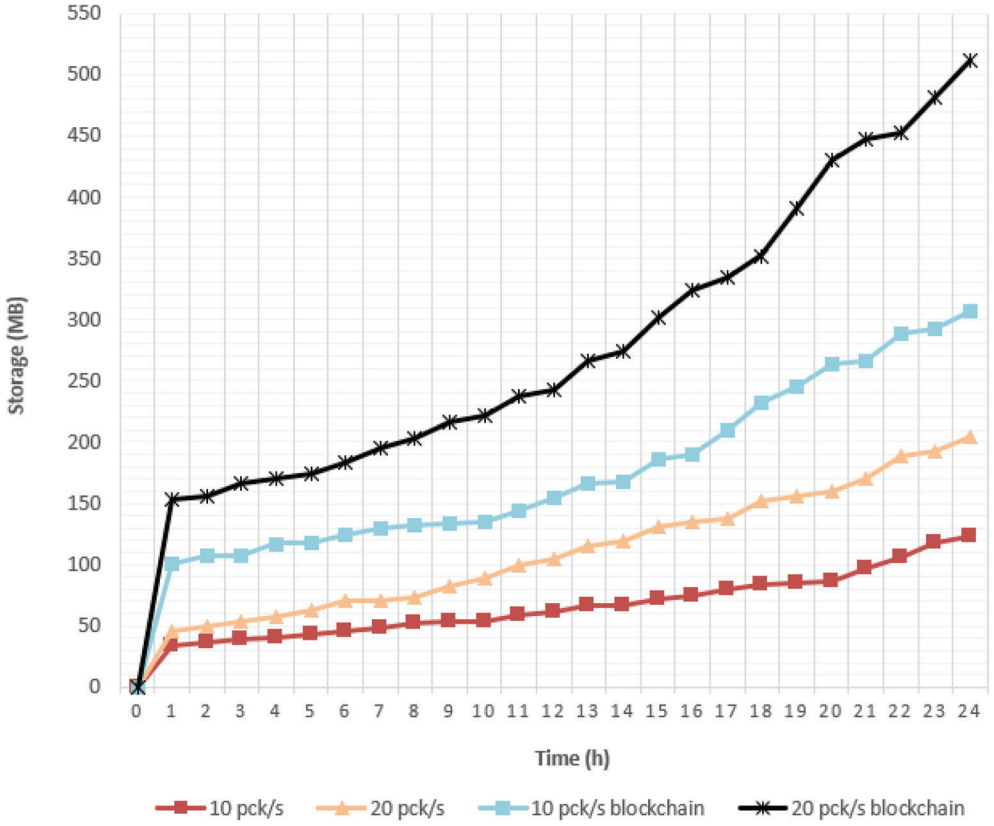

FIGURE 7 Time required for NOSs' storage depletion

change. In particular, from Figure 6, it emerges that a scenario without the blockchain integration and 20 packets/second as the frequency of data fetching requires more memory than a scenario with the blockchain integration and 10 packets/second as the frequency of data fetching. Hence, we can conclude that the integration of blockchain does not affect in a relevant way the storage occupancy on NOSs. In this regard, a certain relevance is assumed by analyzing NOS's storage depletion. To assess such a metric, we tried to stress the system in order to deplete the storage on one NOS totally; to this end, we run the proposed solution in a network composed of a single NOS, with and without the blockchain-related functionalities. The investigated NOS has 512 MB of RAM (i.e., which corresponds to that provided by the Raspberry Pi 2 model), and Figure 7 shows the storage occupancy trend during the time. The outcome reveals that the storage is exhausted after almost 24 h in the heaviest scenario (i.e., data rate equals to 20 packets/second in the presence of blockchain functionalities). Note that such a result is obtained when the considered IoT platform is composed of only one NOS, instead of a network of four NOS (as presented at the beginning of Section 5), thus greatly reducing the distribution of the tasks. Moreover, the policy change rate set to 0.5 requests/second is overstressed in a real scenario. We can conclude that the system is mostly resilient in normal conditions for such a reason. At the beginning of this section, a realistic view of policies changes' frequency is pointed out. A comparison between Figures 6 and 7 further helps in understanding the evolution of memory occupancy during the time in a normal and in an overstressed situation, respectively.

Finally, in case of running out of storage, supposing that we cannot easily replace NOSs, the solution could be the triggering of a blockchain *reset* procedure on the whole NOSs' network. A trusted administrator should be responsible for initiating the new blockchain. Otherwise, a hierarchical blockchain storage structure, supported by a cloud, could be adopted,[39] in order to keep stored on NOS only the more recent blocks of the blockchain, while less current blocks reside on the cloud.

5.2 | Computing effort

The average CPU load on NOSs is measured in two different situations: (i) the IoT system adopting sticky policies only, as presented in Reference 23; (ii) the newly proposed solution with blockchain's integration and policies changes. Figure 8 shows the distribution of the CPU load on the analyzed NOSs, with a variable data rate, while Figure 9 presents the same results in a line graph representation. As expected, the CPU load slightly increases

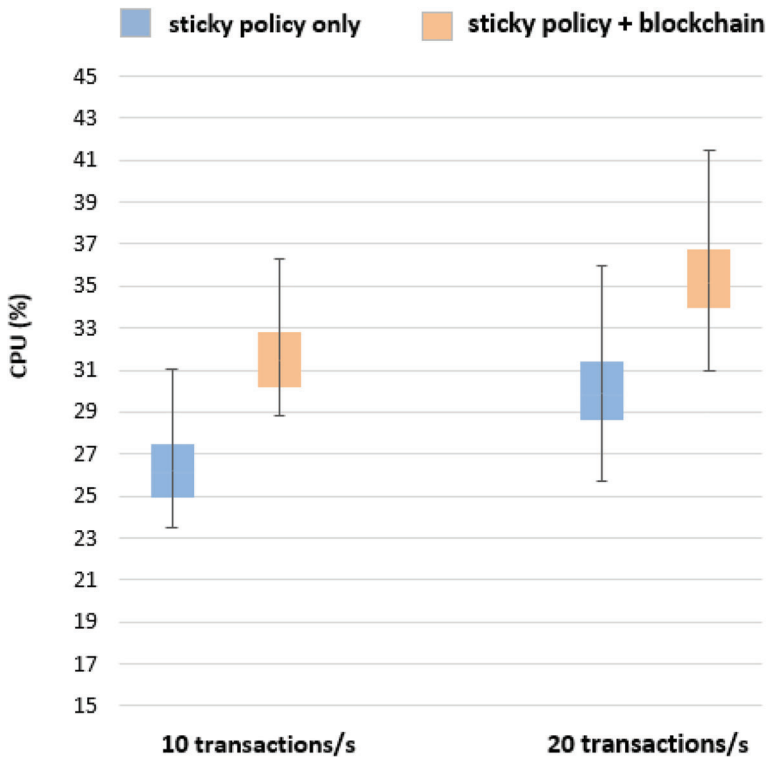

FIGURE 8 Average CPU load on NOSs: Whiskers-box diagram with and without blockchain

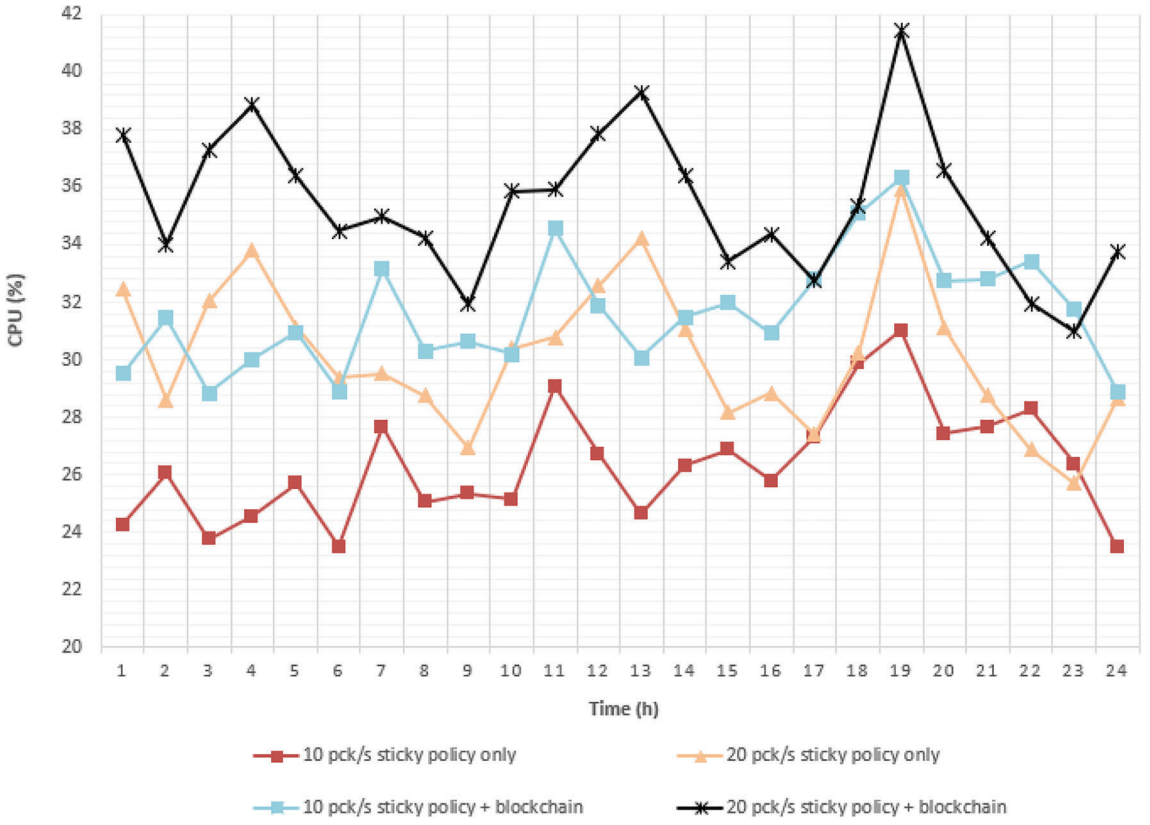

FIGURE 9 Average CPU load on NOSs: Line graph with and without blockchain

in the scenarios with the blockchain integration due to the effort required by the execution of *isChainValid* function (presented in Section 4.1), which is needed to reach the consensus. However, such an increment is not so high because some energy is saved due to the fact that no information exchanges are required with TA, as in the previous version of the platform. In fact, in the new system, TA is no longer needed.

While additional studies, covering larger deployments, are needed, the results suggest that the proposed blockchain approach can actually scale rather well: with a data rate equal to 10 transactions/second, CPU load is below 31.5%, with respect to the 28% of the system without integration with blockchain; increasing data rate to 20 transactions/second, CPU load is still below 36%, with respect to the 32% without the integration with blockchain. Despite the required CPU load increase, the proposed solution provides a high level of security, as previously demonstrated in Section 4.2. Concerning this kind of architecture, there are two approaches to tackling overhead: (i) introducing more NOSs into the network, to better balance the effort spent for data analysis, normalization, topic assignment, and resources' management, in general; (ii) making each NOS more powerful, by incrementing its computational resources.

5.3 | Latency

An important metric to consider is the additional delay introduced by the blockchain mechanism with respect to the previous NOSs' system integrated with sticky policies.[23] The main difference between the two approaches regards the communications among the involved entities. In fact, in the new version of the network, TA no longer exists, and, therefore, the interactions between NOS and TA do not affect the delay of data transmission from their reception to the sharing with the subscribers. However, a new delay is introduced by the blockchain mechanism, which consist of: (i) the generation of the blocks (constructor *Block* in Section 4.1); (ii) the consensus request (function *consensusBlock* in Section 4.1); (iii) the validation of the blockchain (function *isChainValid* in Section 4.1). Moreover, NOSs must exchange information with each other to correctly validate the block, thus generating a new traffic overhead, since the previous version of the architecture did not require communication among NOSs.

In both the solutions, NOSs store sticky policies, and transmit them along with data so that recipients can independently determine if they can access the information or not, based on the assigned permissions; to obtain the access permissions, recipients can subscribe to certain topics and the subscription is accepted only if the request satisfies the requirements established by the policies associated with the data (and established by the NOSs' administrator). If a sticky policy changes for a certain topic, the subscription may become not valid, thus requiring a new subscription. However, such a mechanism is the same in both approaches. Figures 10 and 11 show the comparison of the distribution and the line graph, respectively, of the generated delays. The graphs demonstrate that the new approach presents an increment in the latency, which can be acceptable in this

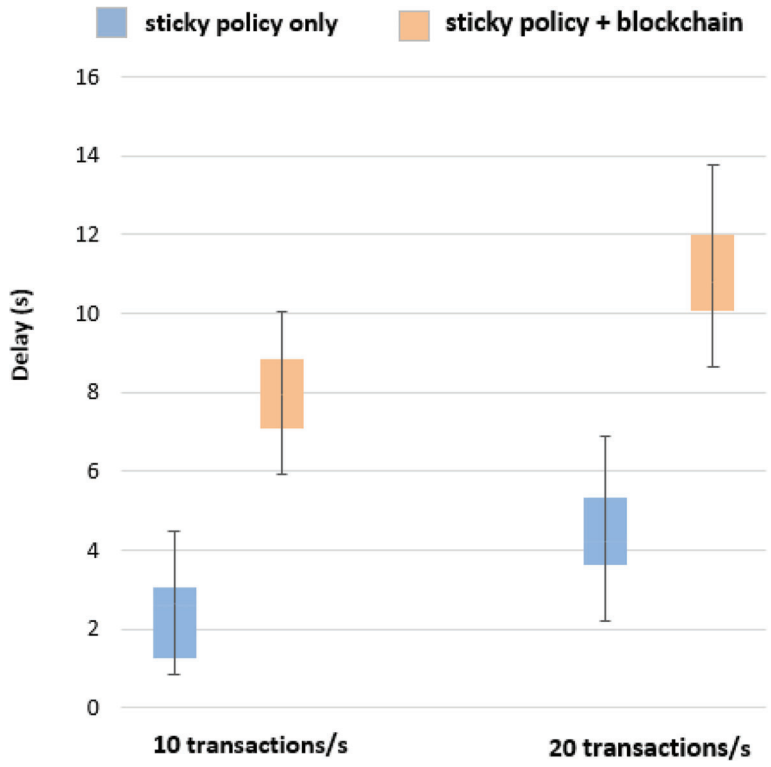

FIGURE 10 Average latency on NOSs: Whiskers-box diagram with and without blockchain

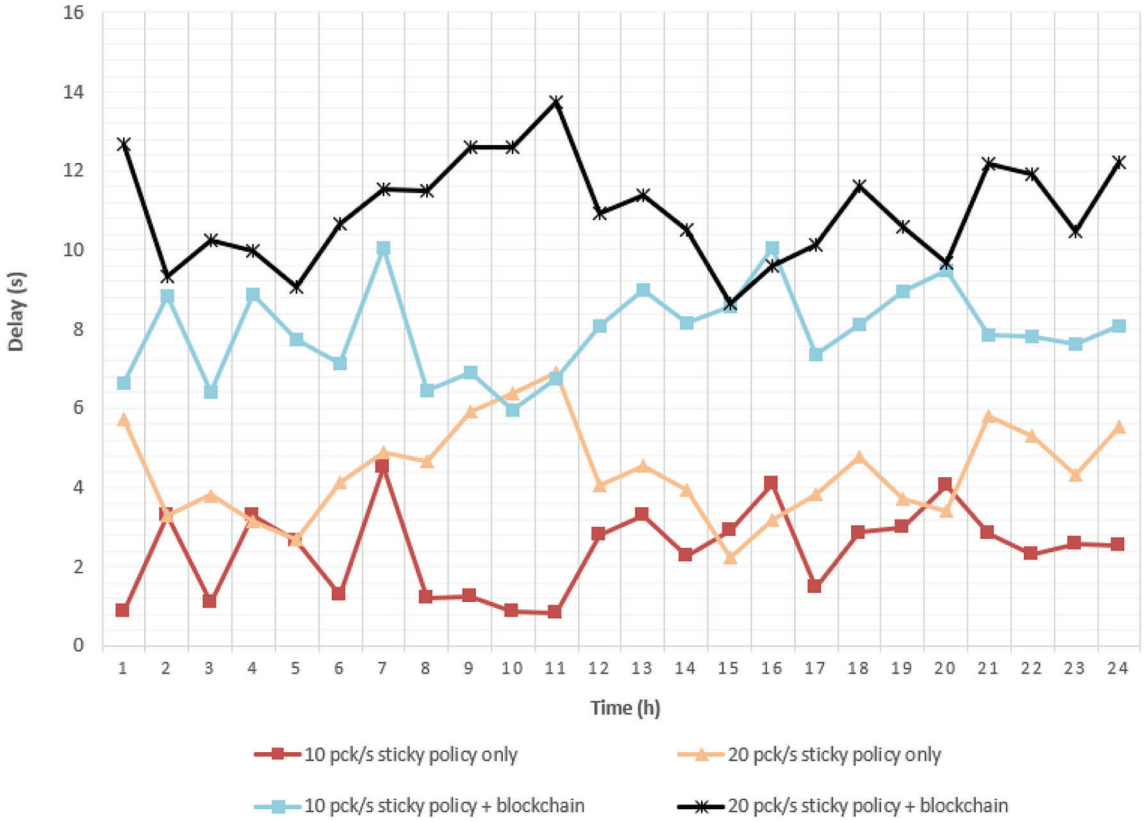

FIGURE 11 Average latency on NOSs: Line graph with and without blockchain

scenario, but should be better analyzed in wider environments, including more NOSs. More in detail, by adopting blockchain, with data rates equal to 10 and 20 transactions/second, latency is below 8 and 11 s, respectively; while, without adopting blockchain, with data rates equal to 10 and 20 transactions/second, latency is below 5 and 7 s, respectively. To cope with the latency issue, we need to lighten communications by reducing the packets' size and the information exchanges. The previous section has just pointed out a partial solution concerning the CPU resources. In fact, by introducing more NOSs or making each NOS more powerful, we can optimize and balance the load on the network, thus reducing the delays.

6 | CONCLUSIONS

The article presented the integration of blockchain technology, and an *honest-but-curious* distributed IoT platform. The work started from the need to find a solution able to guarantee the confidentiality, integrity, and access control of the data transmitted within the IoT network, without the need of any central authority and without considering trusted the IoT platform itself. The need for a trusted authority to release access permissions certainly represents a possible bottleneck, and the vulnerability of policies themselves towards tampering and violations (i.e., in case of malicious components of the IoT platform) is a serious problem. Such issues have been overcome with the introduction of permissioned blockchains, which allow transactions to be managed without considering reliable the IoT platform itself and without requiring any centralization. In this way, the distributed IoT platform is now responsible for the management of the blockchain, thus gaining an important benefit: (i) preserving the resources of the constrained IoT devices, since a fog layer performs the heaviest security tasks; (ii) protecting the policies; (iii) identifying malicious components belonging to the IoT distributed platform.

The designed solution has been validated by means of a simple yet real test-bed, composed of four NOSs' prototypes installed on Raspberry Pi devices. Note that NOS architecture has been chosen for testing purposes. Performance indices like computing effort, storage overhead, and latency have been evaluated. Concerning possible threats, we recognized the robustness of the proposed IoT system with respect to integrity, confidentiality, denial of service, access control, and majority attacks.

As a future development of the presented work, we aim to test the described scenario in a more complex environment, composed a wider number of data sources, and, possibly, simulating malicious behaviors, in order to carry out further experiments about the performance of the proposed approach. Moreover, it would be interesting to integrate the envisioned solution in one of the well-known permissioned blockchain frameworks, such

as Hyperledger Fabric[‖] or Corda[**]. Furthermore, we would like to investigate other blockchain techniques (e.g., consensus algorithms) and evaluate a possible integration in the presented system. Hence, an in-depth analysis of anonymity requirements will also be taken into account in the future.

DATA AVAILABILITY STATEMENT

Data sharing not applicable—no new data generated.

ENDNOTES

[*]https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html
[†]http://nodejs.org/
[‡]http://www.mongodb.org/
[§]http://mosquitto.org
[¶]Node.js v12.4.0, crypto-js library https://nodejs.org/api/crypto.html
[#]http://traces.cs.umass.edu/index.php/Smart/Smart
[‖]Hyperledger Fabric, https://www.hyperledger.org/use/fabric
[**]Corda, https://www.corda.net

ORCID

Sabrina Sicari https://orcid.org/0000-0002-6824-8075

REFERENCES

1. Sicari S, Rizzardi A, Grieco LA, Coen-Porisini A. Security, privacy and trust in Internet of Things: the road ahead. *Comput Netw*. 2015;76:146-164.
2. Liu J, Xiao Y, Chen CP. Authentication and access control in the Internet of Things. Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW); 2012:588-592; IEEE.
3. Bonomi F, Milito R, Zhu J, Addepalli S. Fog computing and its role in the Internet of Things. Proceedings of the 1st Edition of the MCC Workshop on Mobile Cloud Computing; 2012:13-16; ACM, New York, NY.
4. Mouradian C, Naboulsi D, Yangui S, Glitho RH, Morrow MJ, Polakos PA. A comprehensive survey on fog computing: state-of-the-art and research challenges. *IEEE Commun Surv Tutor*. 2017;20(1):416-464.
5. Abbadi IM, Martin A. Trust in the cloud. *Inf Secur Techn Rep*. 2011;16(3-4):108-114.
6. Zheng Z, Xie S, Dai H-N, Chen X, Wang H. Blockchain challenges and opportunities: a survey. *Int J Web Grid Serv*. 2018;14(4):352-375.
7. Dorri A, Kanhere SS, Jurdak R. Blockchain in Internet of Things: challenges and solutions; 2016. arXiv preprint arXiv:1608.05187.
8. Dorri A, Kanhere SS, Jurdak R, Gauravaram P. Blockchain for IoT security and privacy: the case study of a smart home. Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom workshops); 2017:618-623; IEEE.
9. Dorri A, Kanhere SS, Jurdak R. Towards an optimized blockchain for IoT. Proceedings of the 2nd International Conference on Internet-of-Things Design and Implementation; 2017:173-178; ACM, New York, NY.
10. Dorri A, Kanhere SS, Jurdak R, Gauravaram P. Lsb: a lightweight scalable blockchain for IoT security and privacy; 2017. arXiv preprint arXiv:1712.02969.
11. Biswas K, Muthukkumarasamy V. Securing smart cities using blockchain technology. Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS); 2016:1392-1393; IEEE.
12. Sagirlar G, Carminati B, Ferrari E, Sheehan JD, Ragnoli E. Hybrid-IoT: hybrid blockchain architecture for Internet of Things-pow sub-blockchains; 2018. arXiv preprint arXiv:1804.03903.
13. Novo O. Blockchain meets IoT: an architecture for scalable access management in IoT. *IEEE Internet Things J*. 2018;5(2):1184-1195.
14. Ouaddah A, Abou Elkalam A, Ait Ouahman A. Fairaccess: a new blockchain-based access control framework for the Internet of Things. *Secur Commun Netw*. 2016;9(18):5943-5964.
15. Chen E, Zhu Y, Zhou Z, Lee S-Y, Wong WE, Chu WC-C. Policychain: a decentralized authorization service with script-driven policy on blockchain for internet of things. *IEEE Internet Things J*. 2021. doi:10.1109/JIOT.2021.3109147
16. Puri V, Priyadarshini I, Kumar R, Van Le C. Smart contract based policies for the internet of things. *Clust Comput*. 2021;24(3):1675-1694.
17. Conoscenti M, Vetro A, De Martin JC. Blockchain for the Internet of Things: a systematic literature review. Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA); 2016:1-6; IEEE.
18. Christidis K, Devetsikiotis M. Blockchains and smart contracts for the Internet of Things. *IEEE Access*. 2016;4:2292-2303.
19. Fernández-Caramés TM, Fraga-Lamas P. A review on the use of blockchain for the Internet of Things. *IEEE Access*. 2018;6:32 979-33 001.
20. Panarello A, Tapas N, Merlino G, Longo F, Puliafito A. Blockchain and IoT integration: a systematic survey. *Sensors*. 2018;18(8):2575.
21. Dai H-N, Zheng Z, Zhang Y. Blockchain for internet of things: a survey. *IEEE Internet Things J*. 2019;6(5):8076-8094.
22. Wang X, Zha X, Ni W, et al. Survey on blockchain for Internet of Things. *Comput Commun*. 2019;136:10-29.
23. Sicari S, Rizzardi A, Miorandi D, Coen-Porisini A. Security towards the edge: sticky policy enforcement for networked smart objects. *Inf Syst*. 2017;71:78-89.
24. Sicari S, Rizzardi A, Miorandi D, Cappiello C, Coen-Porisini A. A secure and quality-aware prototypical architecture for the Internet of Things. *Inf Syst*. 2016;58:43-55.
25. Rizzardi A, Sicari S, Miorandi D, Coen-Porisini A. AUPS: an open source authenticated publish/subscribe system for the Internet of Things. *Inf Syst*. 2016;62:29-41.
26. Sicari S, Rizzardi A, Miorandi D, Coen-Porisini A. Internet of Things: security in the keys. Proceedings of the 12th ACM International Symposium on QoS and Security for Wireless and Mobile Networks; November 2016:129-133; Malta.
27. Sicari S, Rizzardi A, Miorandi D, Coen-Porisini A. Dynamic policies in Internet of Things: enforcement and synchronization. *IEEE Internet Things J*. 2017;4(6):2228-2238.
28. Pearson S, Mont MC. Sticky policies: an approach for managing privacy across multiple parties. *Computer*. 2011;44(9):60-68.

29. Karjoth G, Schunter M, Waidner M. Privacy-enabled services for enterprises. Proceedings. 13th International Workshop on Database and Expert Systems Applications; 2002:483-487; IEEE.
30. Reyna A, Martín C, Chen J, Soler E, Díaz M. On blockchain and its integration with IoT. Challenges and opportunities. *Future Gener Comput Syst*. 2018;88:173-190.
31. Khan MA, Salah K. IoT security: review, blockchain solutions, and open challenges. *Future Gener Comput Syst*. 2018;82:395-411.
32. Vukolić M. The quest for scalable blockchain fabric: proof-of-work vs. BFT replication. Proceedings of the International Workshop on Open Problems in Network Security; 2015:112-125; Springer, New York, NY.
33. Gervais A, Karame GO, Wüst K, Glykantzis V, Ritzdorf H, Capkun S. On the security and performance of proof of work blockchains. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security; 2016:3-16; ACM, New York, NY.
34. Castro M, Liskov B. Practical byzantine fault tolerance and proactive recovery. *ACM Trans Comput Syst*. 2002;20(4):398-461.
35. Courtois NT, Grajek M, Naik R. Optimizing sha256 in bitcoin mining. Proceedings of the International Conference on Cryptography and Security Systems; 2014:131-144; Springer, New York, NY.
36. Sicari S, Rizzardi A, Miorandi D, Coen-Porisini A. REATO: reacting to denial of service attacks in the Internet of Things. *Comput Netw*. 2018;137:37-48.
37. Barker S, Mishra A, Irwin D, Cecchet E, Shenoy P, Albrecht J. Smart*: an open data set and tools for enabling research in sustainable homes. *SustKDD*. 2012;111:112.
38. Carminati B, Ferrari E, uglielmi M. Secure information sharing on support of emergency management. Proceedings of the 2011 IEEE 3rd International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing; 2011:988-995; IEEE.
39. Wang G, Shi Z, Nixon M, Han S. Chainsplitter: towards blockchain-based industrial IoT architecture for supporting hierarchical storage. Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain); 2019:166-175.

How to cite this article: Rizzardi A, Sicari S, Miorandi D, Coen-Porisini A. Securing the access control policies to the Internet of Things resources through permissioned blockchain. *Concurrency Computat Pract Exper*. 2022;e6934. doi: 10.1002/cpe.6934