



# Blockchain-based authentication and authorization for smart city applications

Christian Esposito<sup>a,\*</sup>, Massimo Ficco<sup>b</sup>, Brij Bhooshan Gupta<sup>c,d</sup>

<sup>a</sup> Department of Computer Science, University of Salerno, Via Giovanni Paolo II 132, I-84084 Fisciano (SA), Italy

<sup>b</sup> Department of Engineering, University of Campania Luigi Vanvitelli, via Roma 29, I-81031 Aversa (CE), Italy

<sup>c</sup> Department of Computer Engineering, National Institute of Technology Kurukshetra Haryana, Frustoo Chowk, NIT, Mirzapur Part, Haryana 136119, India

<sup>d</sup> Department of Computer Science and Information Engineering, Asia University, No. 500 Lioufeng Rd., Wufeng, Taichung 41354, Taiwan

## ARTICLE INFO

### Keywords:

Smart city security  
Access control  
Identity management  
Blockchain  
Decentralized information systems

## ABSTRACT

The platforms supporting the smart city applications are rarely implemented from scratch by a municipality and/or totally owned by a single company, but are more typically realized by integrating some existing ICT infrastructures thanks to a supporting platform, such as the well known FIWARE platform. Such a multi-tenant deployment model is required to lower the initial investment costs to implement large scale solutions for smart cities, but also imposes some key security obstacles. In fact, smart cities support critical applications demanding to protect the data and functionalities from malicious and unauthorized uses. Equipping the supporting platforms with proper means for access control is demanding, but these means are typically implemented according to a centralized approach, where a single server stores and makes available a set of identity attributes and authorization policies. Having a single root of trust is not suitable in a distributed and cooperating scenario of large scale smart cities due to their multi-tenant deployment. In fact, each of the integrated system has its own set of security policies, and the other systems need to be aware of these policy, in order to allow a seamless use of the same credentials across the overall infrastructure (realizing what is known as the single-sign-on). This imposes the problem of consistent and secure data replicas within a distributed system, which can be properly approached by using the blockchain technology. Therefore, this work proposes a novel solution for distributed management of identity and authorization policies by leveraging on the blockchain technology to hold a global view of the security policies within the system, and integrating it in the FIWARE platform. A detailed assessment is provided to evaluate the goodness of the proposed approach and to compare it with the existing solutions.

## 1. Introduction

The Smart City paradigm mainly consists in the application of the ICT technologies to improve the main processes related to the optimal management of crucial aspects of the city governance, to create novel and innovative services by connecting the physical infrastructures of the city with the human, social and intellectual capital living the city (Petrolo, Loscri, & Mitton, 2017). A smart city solution has a layered architecture as in Fig. 1. Typically, this is realized by leveraging the use of different types of electronic sensors and/or citizen empowerment services, to collect several kinds of data, mainly related to pollution, traffic, citizen satisfaction, etc. Such a vast amount of data is further distributed towards the edge and/or cloud for various stages of processing (Ficco, Esposito,

\* Corresponding author.

E-mail addresses: [esposito@unisa.it](mailto:esposito@unisa.it) (C. Esposito), [massimo.ficco@unicampania.it](mailto:massimo.ficco@unicampania.it) (M. Ficco), [bbgupta.nitkr@gmail.com](mailto:bbgupta.nitkr@gmail.com) (B.B. Gupta).

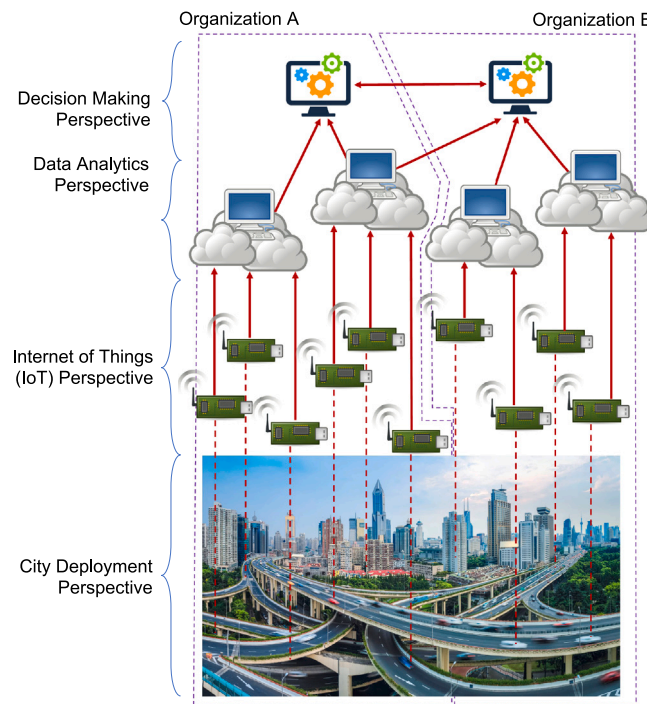


Fig. 1. Scheme of a smart city as the integration of multiple ICT infrastructures and decision makers.

Xiang, & Palmieri, 2017; Mohamed, Al-Jaroodi, Jawhar, Lazarova-Molnar, & Mahmoud, 2017), according to the paradigms of the Internet of Things (IoT) (Du, Santi, Xiao, Vasilakos, & Fischione, 2019). Such processing is needed to infer new knowledge to be used to manage assets and resources efficiently, and to provide feedback on the optimality of the policies put in place within the city. The exploited sensors can be deployed at citizens' devices, public transportation vehicles, and assets with the intent of monitoring principal municipal business process (Petrolo et al., 2017), such as traffic and transportation systems, power plants, water supply networks, waste management, crime detection, information systems, schools, libraries, hospitals, and other community services. This allows city officials and policy makers to interact directly with both community and city infrastructures, and monitor what is happening in the city and how certain phenomena are evolving over time. Such a novel process must allow real-time responses to detect possible issues and put countermeasures against them promptly. Various solutions implement the services required to implement such a vision, such as the well known FIWARE platform (Carnevale, Celesti, Di Pietro, & Galletta, 2018). It offers an OpenStack-based cloud environment plus a rich set of open standard APIs to acquire data from the IoT of the smart city, process and store such data, and provide advanced user interaction.

A smart city solution is not realized from scratch by a single municipality that owns and manages it. Still, it is the product of the integration of some existing infrastructures owned by multiple different organizations that collaborate to make the city smarter, as illustrated in Fig. 1. For example, in a city, the public transportation company may monitor its buses and trains. In contrast, the various private companies related to bike, car, and/or scooter sharing have their ICT infrastructure to control their vehicles and customers. A municipality's central decision-making service can be fed with the monitoring data coming from these infrastructures, jointly with some traffic monitoring sensors deployed along the road by the municipality itself, to have a glimpse in real-time of the traffic state within the city and plan ahead public transportation policies and transportation improvements. Therefore, a federated architecture characterizes the current platforms for a smart city (Ahuja & Wheeler, 2020; Bruneo et al., 2019; Tewari & Gupta, 2020), where the various integrated infrastructures exchange data with the primary decision point of the municipality or even multiple decision points. In fact, there may be multiple decision points in big cities, represented by the various boroughs government that typically compose a city. Therefore, a smart city is an ecosystem of interacting services producing, analyzing and storing data, and making decisions by adjusting the city policies or sending feedback/advice to the citizens to optimize their behaviors.

The application domains for smart city platforms are mainly related to the optimal management of the traffic, waste or energy consumption, but these platforms are progressively being used also in critical cases to realize more secure and safer cities (McClellan, 2019). Examples of these critical cases involve the support of the first responders (police forces, fire brigades, emergency teams etc.) in tracking and preventing malicious people from approaching sensitive places, to rescue citizens in post disaster-scenarios or others. In these cases, the supporting platforms must accomplish a high degree of security as mission-critical data being exchanged and stored, by avoiding unauthorized users to have access to their functionalities and/or data, and to avoid any information leakage. Several approaches have been proposed in order to provide the suitable security and protection degree for smart city solutions (Al-Sharif, Al-Saleh, Alawneh, Jararweh, & Gupta, 2020; Bhushan & Gupta, 2019; Li, Deng, Gupta, Wang, & Choi, 2019). In particular,

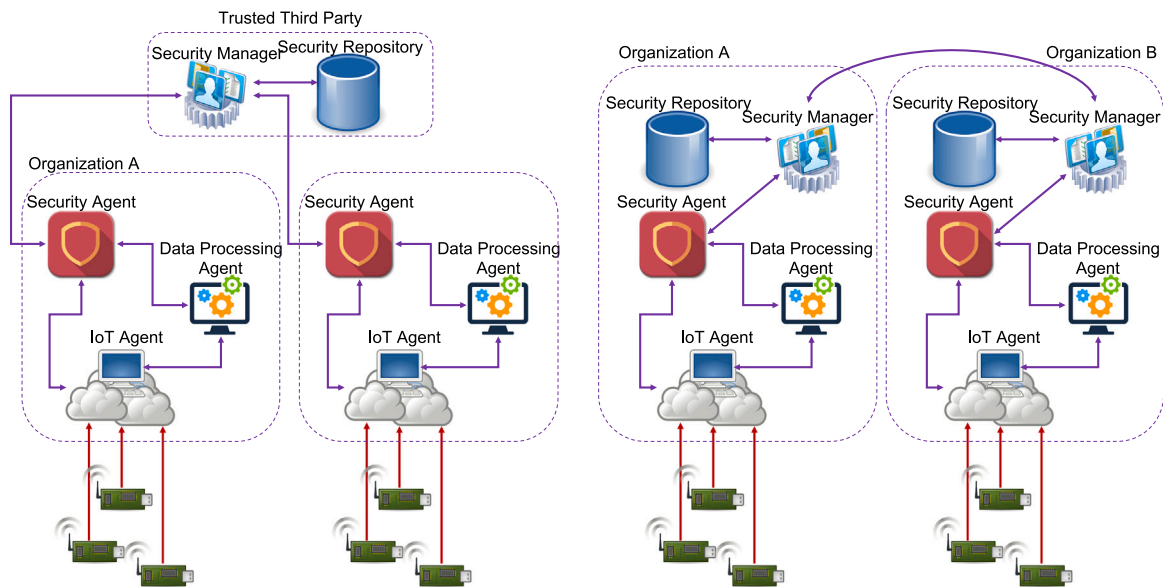


Fig. 2. Security Architecture within the smart city platforms according to a centralized (a) and a federated (b) approach.

this means that proper means to authenticate and authorize users must be implemented within all the main solutions, mainly compliant to the main standard, such as the XACML (Standard, 2005) in the FIWARE platform (Alonso et al., 2019). The drawback of these solutions is their underlying centralized nature where the access control policies and the identity attributes are stored within a single database owned by a trusted third party, as depicted in Fig. 2(a). When a multi-tenant deployment model is applied as in Fig. 1, a centralized solution is not suitable as a single security domain is lacking. In fact, each federated organization may have defined its own security policies' model and a security architecture. Imposing a centralized repository and a single policy model is not suitable but federating the various repositories, so that, if a security claim cannot be locally verified, a distributed query can be performed at the various security managers at the connected organizations. However, this approach is slow and it would be more efficient that each organization will keep locally a copy of all the valid security policies. This, however, implies to resolver the underlying issues of keeping consistent replicas of an object within the context of an asynchronous system, such as the Internet-wide infrastructure of a smart city, and avoid possible vulnerabilities during the update of these objects. The blockchain technology (Monrat, Schelén, & Andersson, 2019) has proved to be useful to deal with this problem (Zahed Benisi, Aminian, & Javadi, 2020), and have found increasing success in its application within the context of the smart cities (Oham, Michelin, Jurdak, Kanhere, & Jhaab, 2021; Xie et al., 2019), smart factories (Putz, Dietz, Empl, & Pernul, 2021) and the IoT (Baniata, Anaqreh, & Kertesz, 2021; Chen, Srivastava, Parizi, Aloqaily, & Al Ridhawi, 2021; Wang et al., 2019), also to design security services (Berdik, Otoum, Schmidt, Porter, & Jararweh, 2021; Li, Wu, Jiang, & Srikanthan, 2021; Salman, Zolanvari, Erbad, Jain, & Samaka, 2019; Zhao, Chen, Liu, Baker, & Zhang, 2021).

This work's contribution is to leverage on the blockchain for the distributed management of identity attributes and authorization policies within the ecosystem, supporting the smart city's realization. This work approaches the security policies' storage with the definition of block structures and smart contracts within a blockchain platform. Such a solution has been integrated within the context of the above-mentioned FIWARE platform, compared with the existing solutions of centralized and federated storage to assess the offered quality in terms of latency in inserting, querying, and altering the stored security-related information.

The paper is structured as follows. Section 2 describes the existing state of the art on authentication and authorization within the context of smart cities and the current literature on using the blockchain within the context of identity management and access control. Section 3 describes in detail the design and implementation of the proposed solution, while Section 4 presents the conducted assessment by illustrating the used testbed, the evaluation criteria, and discussing the obtained results. We conclude with Section 5, giving some lessons learned and a plan for future work.

## 2. Background and related works

Authentication consists of validating the identity and related attributes exposed by a user/machine when requesting to access a given ICT platform's data or functions. Identity-related information is stored and managed by the so-called Identity Management System (IdM) (Miyata et al., 2006). Authorization represents allowing or denying incoming requests according to access control policies issued by the resource's owners regulating its usage. It is implemented in the so-called Authentication and Authorization Service (AAS). Such policies can be represented according to various models, such as based on roles, attributes, and so on (Tourani, Misra, Mick, and Panwar (2017), and exchange protocols and data formats, such as the Security Assertion Markup Language

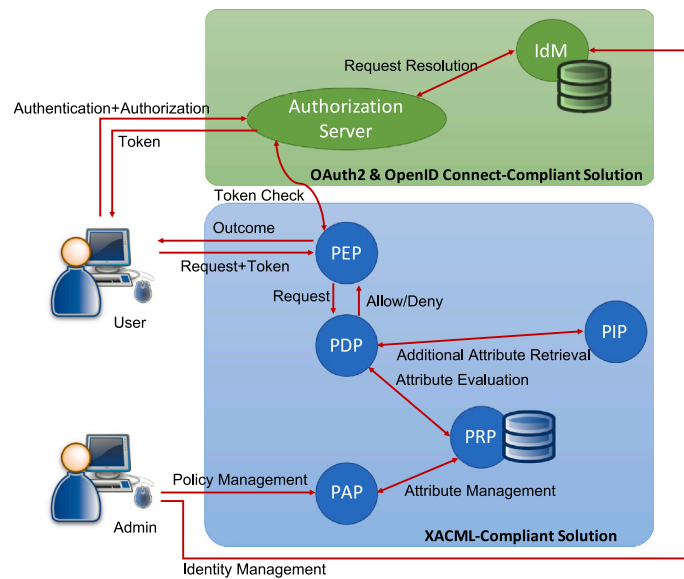


Fig. 3. Schematic architecture of the access control policy verification compliant with the XACML standard, and its possible integration with OAuth2 and OpenID Connect specifications.

(SAML) (Lockhart & Campbell, 2008). While in the beginning, each designer has thought its design for a software solution realizing the IdM and AAS, over time, a series of standards emerged to have a reusable design and solution across all the possible projects.

The most known implementation strategy for designing and implementing an AAS is the eXtensible Access Control Markup Language (XACML) (Standard, 2005). This is an OASIS standard specifying the language to express access control policies and the architecture of a solution to evaluate them as a set of web services. The architecture in assessing access control policies is made of a set of logical components interacting with each other, illustrated in Fig. 3. First, the Policy Enforcement Point (PEP) represents the front end protecting the resources from external requests, extracting the details for the access control verification. It interacts with the Policy Decision Point (PDP) by passing the details extracted from the request and obtaining the decision to grant the access or define it. Such a decision is made by interacting with the Policy Information Point (PIP), acting as a source of attribute values (*i.e.*, a resource, subject, environment) and the Policy Retrieval Point (PRP) responsible for storing and managing the XACML policies. Such policies are administrated by the Policy Access Point (PAP) that received the administrator's requests to insert, alter, or delete XACML policies. In combination with XACML, we can find OAuth2, which specifies the message flows for web applications to realize access control and regulates how the user and the PEP interacts (Hammer-Lahav & Hardt, 2011). Specifically, OAuth2 is an authorization protocol composed of two phases. In the first phase, the user interacts with an authorization server, acting in accordance with the resource owner, authorized to use a particular resource, and receives an authorization token in case of a positive decision. In the second, the user uses the access token to obtain the right to use the intended resource. Such a protocol realizes a delegated access and pseudo-authentication and is complementary to and distinct from OpenID (Bellamy-McIntyre, Luteroth, & Weber, 2011), which is a proper authentication protocol. The main difference between OpenID and OAuth2 is that the IdP, in the first case, returns an assertion of identity for the user, as a result of successful authentication, to perform authorization. In contrast, in the second case, the user obtains an access token to be included in its further requests as proof that he/she has permission from the owner to access the requested resource. As obtaining an access grant may imply the user authenticating him/herself, a successful OAuth access token request is typically mistaken as an authentication method, but this is not fully true. In fact, the access token does not describe the user's identity, and if the token is compromised, it can be easily used for running impersonation attacks. As the assertions of identity returns by OpenID are equipped with a cryptographic verification mechanism, its use prevents the possibility of having such attacks possible. Recently, an authentication layer compliant with OpenID and built on top of the OAuth 2.0 authorization framework has been proposed and named as OpenID Connect (Sakimura, Bradley, Jones, De Medeiros, & Mortimore, 2014), where an ID token with identity claims is returned in addition to the access token. XACML, OAuth2, and OpenID Connect are typically integrated, as represented in Fig. 3, to have a complete IdM and AAS realization. Interested readers can refer to the various existing surveys for more details on these concepts and technologies, such as Bertin, Hussein, Sengul, and Frey (2019), Paci, Squicciarini, and Zannone (2018), Tourani et al. (2017) and Zhang et al. (2018).

The policy storage in the PRP, as well as the repository of identity attributes hold by a IdM, are typically realized utilizing a relational database when a single root of trust is available in a system. When such centralized management of policies is not suitable due to the federated deployment applied to the system of interest or the multi-tenant model, more advanced solutions are needed, such as a federation of databases. Specifically, multiple databases are available in the system, one for each organization, and a proper update and query procedure across them, such as the Three-Phase Commit (3PC) for the atomic update of data split across the databases (Mohan & Lindsay, 1985) or federated queries (Tan, Chirkova, Gadepally, & Mattson, 2017). In the context of the

**Table 1**

Main smart city platforms and their design choices for the AAS and IdM.

Ref.	Solution	AAS solution	IdM Solution	Centralized	Multi-Tenant
Carnevale et al. (2018)	FIWARE	XACML	OAuth2.0	X	X
Perera, Zaslavsky, Christen, Compton, and Georgakopoulos (2013)	OpenIoT	Custom	OAuth2.0	X	
Bauer (2017)	OpenRemote	Custom, Keycloak		X	X
Enterprise (2016)	Kaa IoT	Custom, Kaa Tekton	OAuth2.0	X	X
Del Esposte, Kon, Costa, and Lago (2017)	InterSCity	Custom, Kong's API Gateway		X	
Gracia (2018)	Sofia4Cities	XACML	OAuth2.0	X	
Moskvitch (2016)	Sentilo	Custom, Token Based Authentication Server		X	
Khare, Merlino, Longo, Puliafito, and Vyas (2020)	SmartME	Custom, NGINX reverse proxy server		X	
Sanchez et al. (2014)	SmartSantander	Custom, web-based Access Control Interface		X	

open-source smart city platforms, effortless approaches have been typically used without compliance with one of the widely known standards, as listed in Table 1. From such a table, we can notice that the traditional database-oriented storage of identity attributes and access control policies is still considered valid and mature. At the same time, less attention has been devoted to experiment and implement alternative solutions. Greater attention has been paid to the compliance of the implemented solutions to the legal framework for data protection or digital identities, such as in Alonso et al. (2019) where a solution compliant to the electronic identification and trust services (eIDAS) regulation has been described and implemented. This is related to the old idea that the city municipality has to be in charge of acting as the single root of trust and hosting a database with the needed information for authentication and authorization. However, this is progressively changing as the novel concepts of a smart city as an ecosystem is paving the way. In addition, when a multi-tenant deployment is supported, using federated repositories is the preferred solution, but it imposes several limitations.

All of them assume that all the repositories share the same format and model of the hold authorization policies and identity attributes. A solution to this problem exists within the current literature, as it contains means to provide syntactic and semantic interoperability among security policy and attributes (Esposito, 2018). However, having federated repositories causes underlying delays when updating and querying the stored policies and attributes, as the right location of the data of interest needs to be located. As these platforms require fast responses to deal with a request to the AAS, it would be better to copy the global view of the security policies and attributes locally to the server dealing with the user request. However, the consistency among all the data replicas within a distributed asynchronous system is not guaranteed, as the Internet does not have predictable communication delays. As a widely-known result from the theory of distributed systems, consistency, which can be traced back to a consensus problem, is not guaranteed in an asynchronous distributed system (Borowsky & Gafni, 1993). Besides, in such an approach, the possibility of attacks against the replicated repository is neither internal nor external adversaries. It is crucial to guarantee that stored data are hard to alter. Finally, identity attributes and authorization policies are always kept in different repositories, mainly managed by other administrators. It is more convenient to hold them in the same repository to efficiently retrieve them when needed to validate a joint authentication and authorization request, as required by OpenID Connect.

The blockchain (Li, Jiang, Chen, Luo, & Wen, 2020; Nguyen & Kim, 2018) is a promising technology (Wang, Zhang, & Zhang, 2018) consisting of grouping data in blocks that are linked among each other with the hash of the previous blocks, and having participants to the blockchain agreeing on the new blocks to include within the blockchain, by running a distributed consensus algorithm tolerant to crash and byzantine failures. Such technology is used to guarantee consistency of replicated data in a distributed environment, even if being asynchronous, thanks to the used consensus algorithms, and to provide data immutability from intentional and unintentional manumissions thanks to the used crypto primitives. The blockchain has started to be used within the context of the Internet of Things and the smart city for the protection of data (Biswas & Muthukkumarasamy, 2016; Esposito et al., 2018; Wang et al., 2019), in many different applications, in particular to device security-related services, such as access control and authentication. Within smart cities, the available academic publications are mainly theoretical and at the architecture level, with few concrete implementations. On the contrary, the literature within the context of the Internet of Things is more mature and encompasses various solutions and implementations. Interested readers can refer to those two surveys providing a literature analysis on the use of blockchain for storing identity attributes (Zhu & Badr, 2018), and access control policies (Riabi, Ayed, & Saidane, 2019) within the context of the Internet of Things. On the first hand, blockchain is used to hold identity-related data, as in Liu et al. (2017), to substitute the relational repository, and realize self-sovereign identity where user have direct control of their identity attributes and their use. Some of those available solutions also introduce privacy-preserving features, as in Shao et al. (2020), by encrypting personal information hold by the blockchain or anonymizing them. On the other hand, there are two classes of approaches. The first class includes solutions storing access control policies as transactions within the blockchain, as in Ouaddah, Abou Elkalam, and Ait Ouahman (2016), so that new types of transactions have been introduced to grant, get, delegate, and revoke access. The second class includes solutions where access control policies are modeled as smart contracts, as in Novo (2018), so that the users' claims are used as input to activate one of the valid smart contracts and obtain as output the token for the OAuth2 authorization. As a side research effort, blockchain has also been extensively used within trust management to realize dynamic authentication of users and devices within the Internet of Things (Cinque, Esposito, Russo, & Tamburis, 2020; Hammi, Hammi, Bellot, & Serhrouchni, 2018), where trust degrees are stored in the blocks.

By looking at the main related works, we can notice that none of the proposed blockchain solutions have been applied to existing open-source platforms for smart cities, lacking to realize a real transfer of knowledge and possible application to running



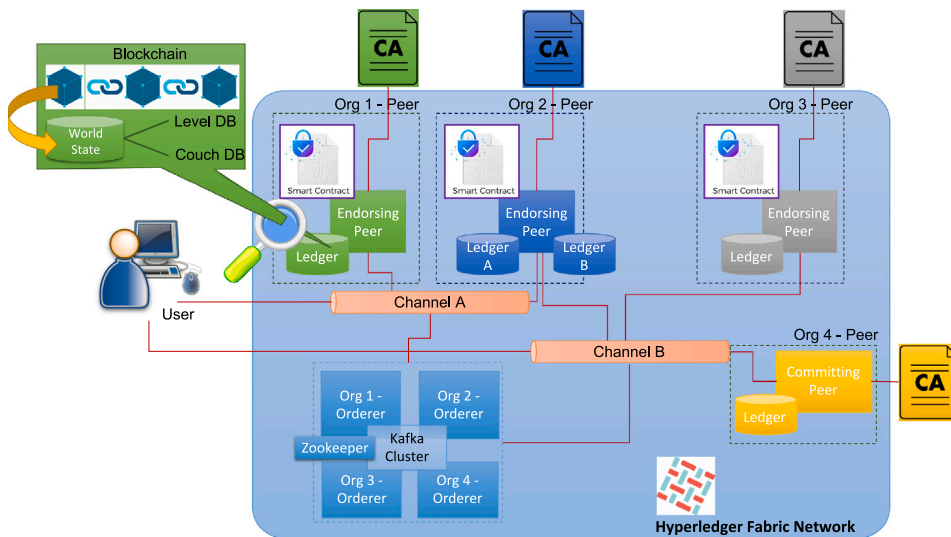


Fig. 4. Architecture of Hyperledger Fabric.

deployments. For this reason, the presented work aims at integrating the blockchain-based solutions within the context of the widely-known FIWARE platform. Solutions are typically designed for access control or identity management, while few of them consider both. This work aims at modeling both kinds of data within the block structure of a blockchain, following the data model envisioned in the repositories of FIWARE, so as to allow a simple and seamless substitution of the existing solutions. In both cases, the adopted approach is the transaction-based, without exploiting the smart contracts. The aim is to use the blockchain as a mere distributed data repository, leaving the implementation of the authentication and authorization logic externally to the blockchain, as provided by FIWARE.

### 3. Design and prototype

The driving idea of providing a decentralized OAuth2-based authentication and authorization solution within the context of the FIWARE platform for realizing a smart city is to substitute the centralized storage of security claims within a database by adopting a blockchain platform to have multiple instances of the knowledge base being distributed and consistent among the various organizations. Such a solution does not require single access control and identity model, as numerous models can easily coexist within the blockchain. The FIWARE platform embodies a modular architecture where Generic Enablers (GE) are software components exposing an interface of RESTful web services, implementing a particular business logic, and being able to be composed. The Orion Context Broker component makes the orchestration and data sharing among the GE and the context capture and distribution. This work aims to design the changes to the security-related FIWARE GE with proper solutions implemented by using a blockchain platform, and the following subsections describe such changes. The second subsection illustrates the extension of the FIWARE Keyrock IdM by introducing the blockchain. In contrast, the last one shows the integration of a custom policy storage based on the blockchain in the FIWARE AuthzForce Server. The next subsection introduces the used blockchain and motivates its selection.

#### 3.1. Selection of the blockchain platform

Different criteria are used to group and classify the various existing blockchain platforms. The most known one considers the openness nature of the node participation to the consensus and the platform's main services. If any node can join the platform and participate in the consensus, the platform is said to be permissionless. If only certain nodes are authorized to participate in the consensus, the platform is permissioned. The different openness characterization of these two groups implies another declination of the famous CAP theorem within the context of the blockchain (Gilbert & Lynch, 2012). In a distributed system connected by the Internet, it is possible to guarantee only two properties among Consistency, Availability, and network Partition tolerance. The remaining property is only eventually achieved with weakened guaranteed. In a distributed system, network partition tolerance must have property, so only one can be guaranteed among the remaining two while the other is weakened. The permissionless platforms, such as Bitcoin and Ethereum, apply various Nakamoto consensus variations and the Proof-of-Work membership policy to promote honest behaviors (Wang et al., 2019). Such an approach allows forks within the blockchain, representing periods where the consistency is not reached and resolved later. Forking is necessary to provide higher availability at the expense of consistency, which is eventually obtained when the fork occurrences are resolved by considering the longer branch (Xiao, Zhang, Lou, & Hou, 2020). This makes permissionless blockchain platforms being examples of an AP system. On the contrary, the permissioned platforms embody strong consensus algorithms, originated from the State Machine Replication (SMR) and Byzantine

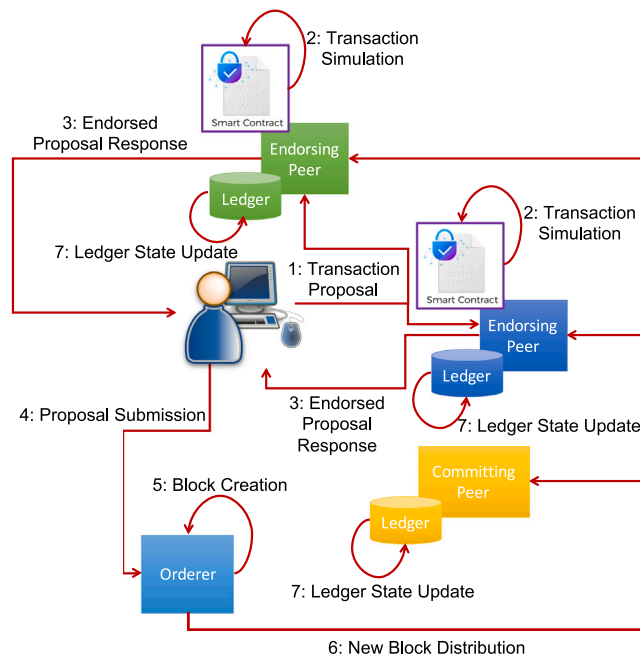


Fig. 5. Sequence of the interactions among the Fabric elements for consensus on new blocks to be added in the chain.

Fault Tolerant (BFT) consensus (Wang et al., 2019; Xiao et al., 2020), and supporting strong consistency among the state hold of the various blockchain replicas without any fork. This means that permissioned platforms represent examples of the CP systems. It is imperative to have consistency among the various repositories holding security policy and attributes within the security systems context, making the CP systems and the corresponding permissioned blockchain platforms more suitable to be integrated within the FIWARE platform. Hyperledger Fabric (Androulaki et al., 2018) is the most known permissioned blockchain platform and has been used within this work. It belongs to a larger open-source project supporting the collaborative development of distributed ledger technologies. It represents a framework for permissioned private blockchain, where only specific nodes are authorized to invoke precise functionalities by using access control lists (ACLs). Specifically, clients interact with Fabric by managing a series of resources, such as user/system chaincodes. Such interactions are granted if they satisfy one of the policies stored within the framework.

The architecture of Fabric is represented in Fig. 4, where the key elements of such a framework are illustrated. The key communication element is represented by the channels that connects all the components within an organization or spanning across organizations. In the case of organization subscribing to multiple channels, they hold as many ledgers as the number of joined channels (as for organization 2 in the figure). All the components underlying the same organization hold a consistent view of their associated ledger. Several components are constituting Fabric architecture. First, there are peers, which are elements storing all transactions on a joining channel with a separate ledger to keep them confidential and share them to only permitted participants on a certain channel. There are two kinds of peers. The endorsing ones that host smart contracts or chaincode simulate the result of the chaincode execution according to an endorsing policy and return the result to the requester. Both the endorsing and committing peers hold a local copy of the assigned ledgers to keep updated based on the incoming new block requests. The ledger is made by a chain of blocks linked utilizing cryptographic hashing functions (the  $i$ th block stores the hash of the  $(i - 1)$ -th, with the only exception of the first or genesis block), while the world state is a consistent view of the variables for the given blockchain, accessible employing the LevelDB (which is a default key-value database) or CouchDB (which is a JSON-based database for richer querying operations). Within such an architecture, multiple Certification Authorities (CA) may be deployed one per each organization or a single centralized server for all of them, with the duty of implementing authentication and authorization of users when interacting with the blockchain through X.509 standard certificates to represent permissions, roles, and attributes for each user. The orderer is the node, or a set of nodes, responsible for receiving the requests of new transactions from the clients, order them, creating new blocks where storing multiple transactions, and distributing them among the peers. In the case of a cluster of orderers needed to provide scalability and fault-tolerance, a Kafka cluster, and a ZooKeeper ensemble are used to manage them. The consensus protocol to add a new block to the blockchain is implemented when the orderer interacts with the peers. In Fabric, a gossiping algorithm is employed, which is known to reach consensus in a probabilistic manner (Esposito, Castiglione, Palmieri, & Ficco, 2017).

The interaction among these components is depicted in Fig. 5. In the beginning, the client submits a transaction proposal to the endorsing peers by using the channel to which it has joined (interaction 1). These peers simulate the transaction result by executing the chain code (interaction 2) and sending back such a response as an endorsement to the client (interaction 3). The transactions and the relative endorsed proposal responses are collected and finally sent to the orderer by the client (interaction 4). The order node receives all the clients' messages, orders the contained transactions, verifies the relative endorsed proposal responses, and creates

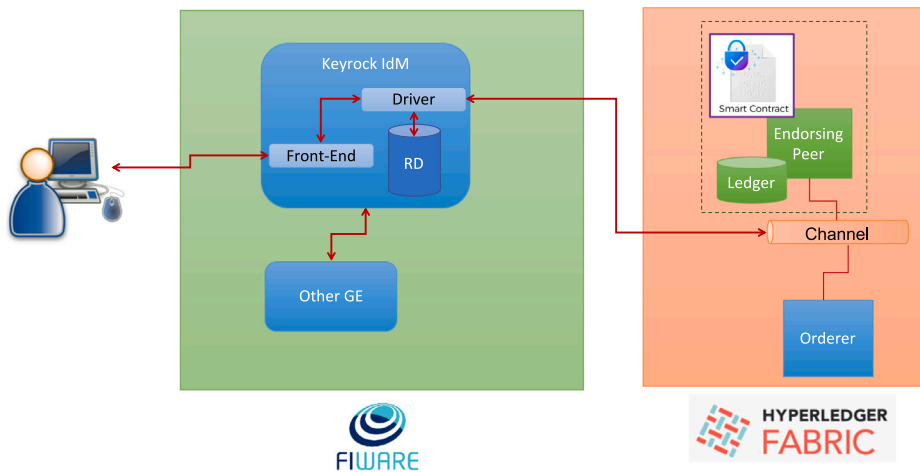


Fig. 6. Integration of the Blockchain in Keyrock.

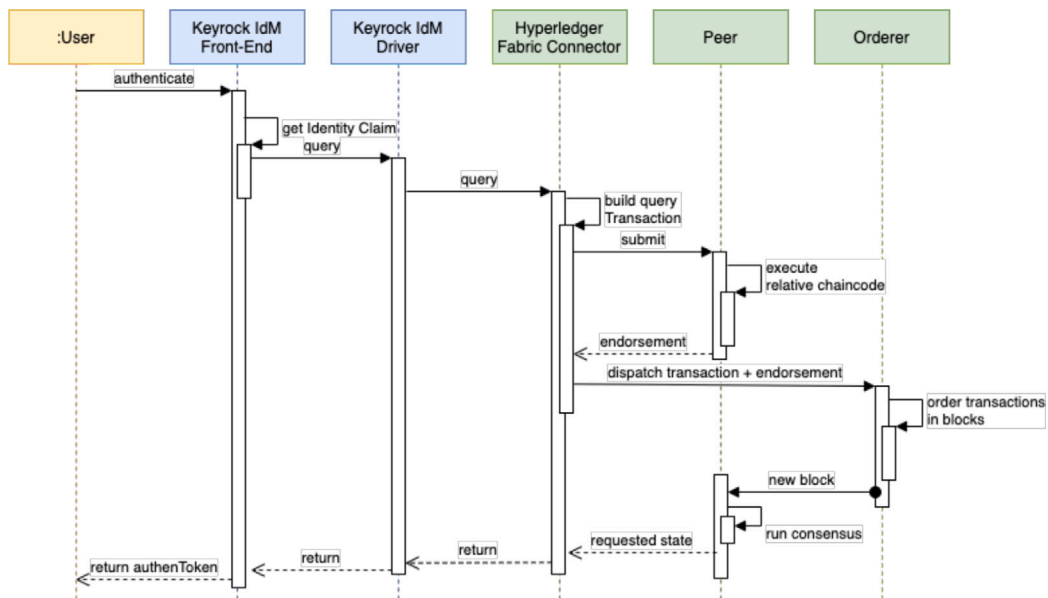


Fig. 7. Sequence Diagram of the authentication validation and identity token return.

new blocks with legitimate and valid transactions (interaction 5). Such blocks are distributed towards all the peers on the joined channel (interaction 6), which runs a chosen consensus protocol and agrees on the new state of the local ledger, which is updated accordingly (interaction 7).

### 3.2. Blockchain-based identity management

Keyrock GE implements the IdM for the FIWARE platform that is compliant with the OAuth2 standard (Hardt et al., 2012). Basically, users can register themselves in such a service and their applications, with relative resources. An organization is a set of users sharing resources of their applications. Such roles and permissions are manageable by this server's proper API, compliant to the OAuth2 protocol. Users can use such a GE and other GEs (such as the PEP Proxy Wilma or the AuthZForce PDP GE), to submit OAuth2 requests and obtain an authentication token, to be included in HTTP headers for user authentication purposes when requesting resources, or to register/modify roles and permissions. The details on users, applications, and resources are stored within a proper local relational SQL database, implemented with MySQL. The driver forwarding the data update and query operations



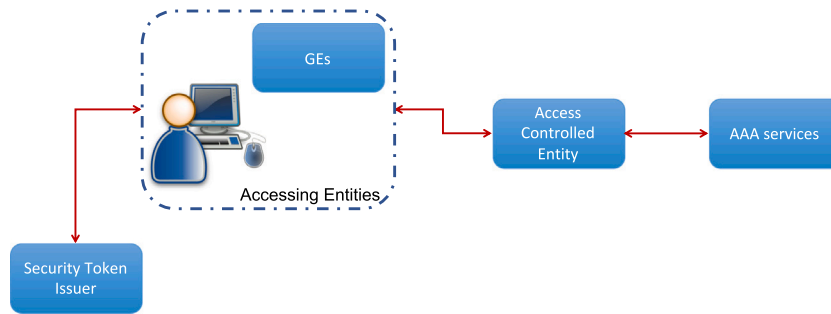


Fig. 8. SENSEI approach for authentication and authorization by means of tokens.

towards the DBMS has been extended by intercepting such operations and redirecting them towards a peer and orderer of Fabric, as illustrated in Fig. 6. A smart contract has been defined to realize the CRUD operations related to active users' details to bring within the blockchain the data stored in the database used by Keyrock. A chaincode has been implemented for the registration, query and modification of identity information, as modeled within the context of the IdM in FIWARE. A JSON object containing name, surname, email, and password is exchanged between the two components in Fig. 6, and integrated within the blocks stored within the peers of Hyperledger Fabric and managed by the chaincode, according to the interaction described in Fig. 7. Specifically, when the user needs to be authenticated, a request arrived at the Keyrock IdM Frontend, which formulates proper predicates to validate such a request. The predicate is passed to the Driver, which forwards it to the DB in its current implementation within FIWARE by using the JDBC driver and library to interact with MySQL. However, in our enhancement, such a query arrived to the Connector we have implemented to mediate with the Fabric blockchain and its peers. With such a predicate, a proper query transaction is built as a JSON Object to invoke the specific function of the chaincode, called with the parameters provided by the user to the Keyrock IdM Front-end. Such a transaction is given to the Peer to have an endorsement token. The transaction and the endorsement token are jointly sent to the Ordered, which aggregates it with other transactions to have a new block distributed to all the peers. These peers execute the selected consensus algorithm, and the one closer to the Connector returns the outcome of the query to the global state of the blockchain. Such a result is returned towards the Keyrock IdM Front-end by following the function invocation list in reverse order. The front-end receives a result syntactically similar to the one expected from the DB in the current implementation. At this time, the front-end's existing logic can run with no changes and return a successful authentication token or an error if the request is legitimate or not. The same interaction is followed if the identity details need to be updated, but the interaction initiator is not a user but an organization's administrator.

### 3.3. Blockchain-based access control

The access control approach adopted by the FIWARE platform is illustrated in Fig. 8, according to the European project SENSEI (Presser, Barnaghi, Eurich, & Villalonga, 2009).<sup>1</sup> In the figure, we can distinguish between the Accessing Entity (AE), which is an user or GE requesting to interact with or use the resource hold by another entity in the infrastructure, and the Access Controlled Entity (ACE), which is the critical GE under the protection by the Authentication Authorization and Audit (AAA) service. The intention is to grant access only to those legitimate AE authorized to interact with and/or use a given ACE resources. For these purposes, the AE authenticates itself by interacting with a Security Token Issuer (STI), providing some identity and security claims, and having back a token to be inserted to any requests to be sent towards an ACE. When receiving such a token, an ACE can interact with AAA services that decide to grant or deny the relative request.

OAuth 2 is the reference standard for implementing the STI, within which the IdP is a key element for the authentication of AE. The AAA services in the platform is mainly offered by the Access Control GE, which is only a consumer of OAuth tokens by (i) validating them, (ii) extracting attributes to be match against the stored access control policies, and (iii) returning an authorization decision to the ACE. Such a GE has been devices according to the XACML standard (Standard, 2005) issued by the OASIS consortium to architect the access control implementation within the context of the Service-Oriented Architecture (SOA) as the interaction of different key components realized as web services. First, a PEP, implemented by the Wilma GE, receives the authorization request from the AE, verifies its integrity and validity, extracts the key information of interest from it, and translates them within the internal dialect adopted among the XACML components. PEP interacts with the PDP, implemented by the AuthZForce GE, responsible for evaluating the requests by considering the active policies stored within the system, determining a decision (Permit/Deny/Not Applicable/Indeterminate), and returning a response to the PDP. The policies are managed by a PRP to lookup, as well as to store XACML policies by using an internal/external database or even flat files. The insertion, modification, or deletion of system administrators' policies are made possible thanks to a PAP. Sometimes the attributes extracted by the PEP from a request may not be sufficient to make a decision, so the PDP may also interact with a PIP to retrieve more detailed information useful to make a decision and to support the queries within the PRP.

<sup>1</sup> [www.sensei-project.eu/](http://www.sensei-project.eu/).

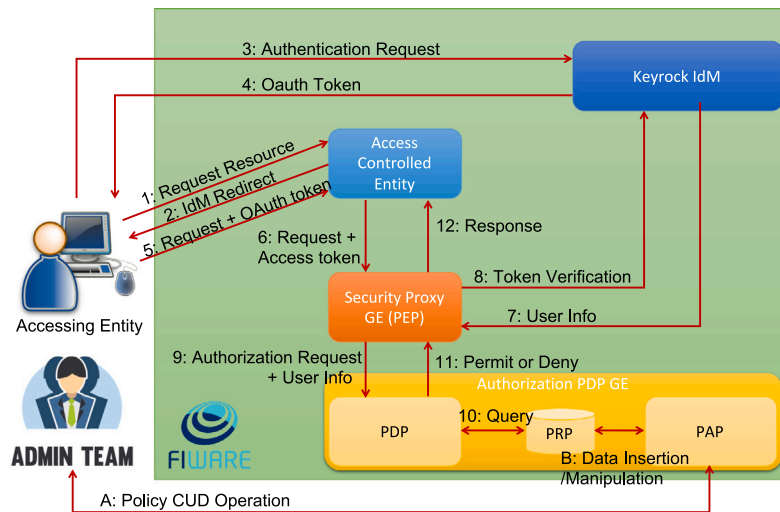


Fig. 9. XACML-based authorization within FIWARE.

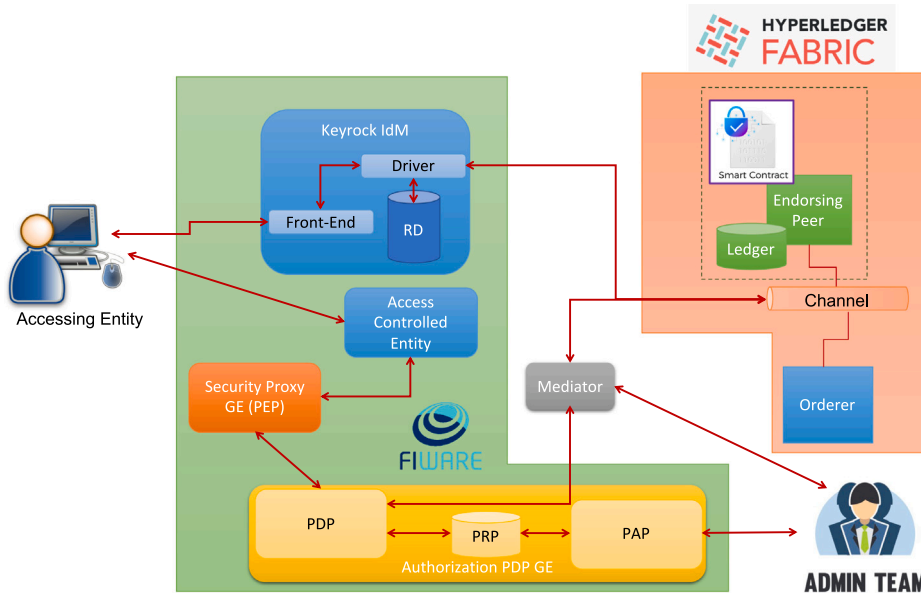


Fig. 10. FIWARE authorization GEs extended by using Blockchain.

Fig. 9 shows how XACML-based architecture has been realized within the context of the FIWARE platform. First, the AE contacts an ACE. If an OAuth token is not associated with such a request, the ACE replies redirecting the AE to the Keyrock IdM to obtain such a token. The AE sends an authentication request to the Keyrock IdM, with the needed user information and attributes, obtaining the OAuth token as a response if the provided information is correct and valid. Then, the client re-sends the request by including the token, which is passed to the Security Proxy GE (realized by the Wilma GE in FIWARE implemented in Node.js) within a proper request. Such a GE acts as the PEP and passes the token to the Keyrock IdM to verify its validity and integrity and obtain the user details. Such an obtained response is used to fill an authorization request to be sent to the Authorization PDP GE, which implements internally a PDP for the evaluation of such requests and the reply with a proper decision, and a PAP for offering to Create Update and Delete (CUD) operations for XACML policies hold by a PRP, consisting into a database. The PDP is implemented using the Authzforce code as a RESTful service implemented in Java by using the Springboot framework contained within Apache Tomcat Server. Specifically, such a code realizes a Policy Provider to make lookup queries in the PRP, which by default, consists of the MongoDB, where XACML policies are stored in a JSON format.

Fig. 10 presents how we have integrated a blockchain solution within the Access Control GEs of FIWARE. The starting point consists in taking a PolicyProvider and integrating a custom driver to send all the lookup requests towards the blockchain instead

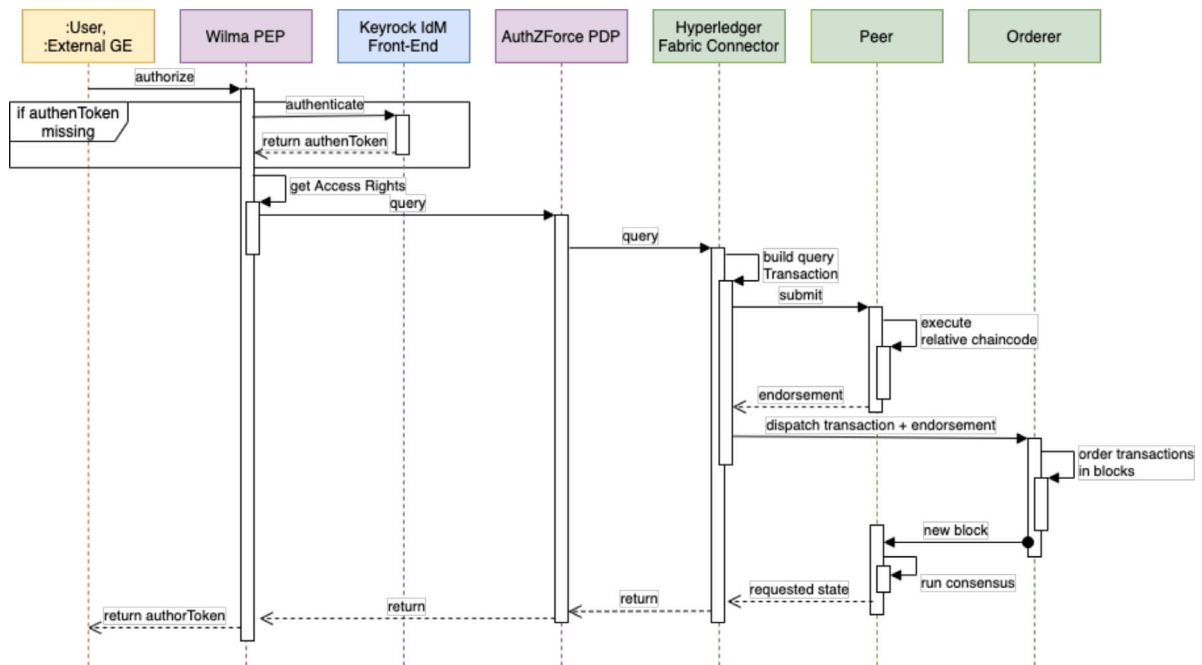


Fig. 11. Sequence Diagram of the authorization validation and access token return.

of MongoDB. As the API of Hyperledger Fabric is in Node.js, while the PolicyProvider code has been written in java. We decided to realize a mediating server and use a socket in the localhost to communicate the custom driver with such a server. Unfortunately, it has been impossible to do the same with the PAP GE provided by FIWARE, as its business logic was strongly coupled with the DB interaction, so administrators have to directly interact with our server for requesting CUD operations of the blockchain. As described in the previous subsection, we have mapped the database's content within the blocks of Fabric and defined four smart contracts to initialize the chains, run query operations, and insert/modify policies. Fig. 11 depicts the interactions among the existing GE and our extension for the query of the authorization policies, while the their administration do no involve PEP and PDP, but is directly between the Admin and External GE with the Hyperledger Fabric Connector. At the beginning, the present of the authentication token is checked, and if it is found missing, the Keyrock IdM Front-End is contacted to obtain it. Afterwards, the access attributes are extracted from the request and a query is sent to the AuthZForce PDP, which does not contact the DB but the Hyperledger Fabric Connector, which performs the query on the blockchain by invoking the proper method of a deployed chaincode and obtains a result. Such a result is passed back to the PEP so as to return an authorization token to the user or requesting GE.

### 3.4. Qualitative security analysis

By looking at the security of the three different solutions, we can notice that Keyrock interacts with its local and federated SQL DB by means of HTTPS and SSL Certificates, which are known to be vulnerable to external attacks, such as an active Man-In-The-Middle attack<sup>2</sup> or a honeypot,<sup>3</sup> and also to inside attacks, as a malicious/compromised administrator is able to compromise the stored data without leaving no trace. The AuthzForce GE uses MongoDB with similar security means by TLS/SSL certificates, which is characterized by a set of vulnerabilities,<sup>4</sup> and similar insider treats of MySQL. On the contrary, the blockchain makes immutable and traceable any past agreed state update (realizing auditing by registering the data provenance), making possible to detect and undo malicious modifications to the security-related data. As, when data are changed, all the participants see such a change and can act to correct it. Also, Hyperledger Fabric has a non negligible attack surface, as described in Dabholkar and Saraswat (2019), which has not been analyzed in details. However, Hyperledger Fabric is able to provide a high security degree thanks to the use of PBFT and Sieve, enhanced by strong identity management and privacy features. The benefit of our blockchain-empowered solution is the removal of the single-point-of failure represented by the DB, even in the federated deployment, as a piece of data of interest (such as an identity attribute or an access policy) is only stored at a single DB instance, whose failure makes such data unavailable. On the contrary, with blockchain each organization has its own replica of the data, or even more than one, achieving a high data availability.

<sup>2</sup> <https://www.idontplaydarts.com/2015/03/mysql-with-ssl-does-not-protect-against-active-mitm/>.

<sup>3</sup> <https://news.sophos.com/en-us/2019/05/24/gandcrab-spreading-via-directed-attacks-against-mysql-servers/>.

<sup>4</sup> [https://www.cvedetails.com/vulnerability-list/vendor\\_id-12752/product\\_id-25450/Mongodb-Mongodb.html](https://www.cvedetails.com/vulnerability-list/vendor_id-12752/product_id-25450/Mongodb-Mongodb.html).

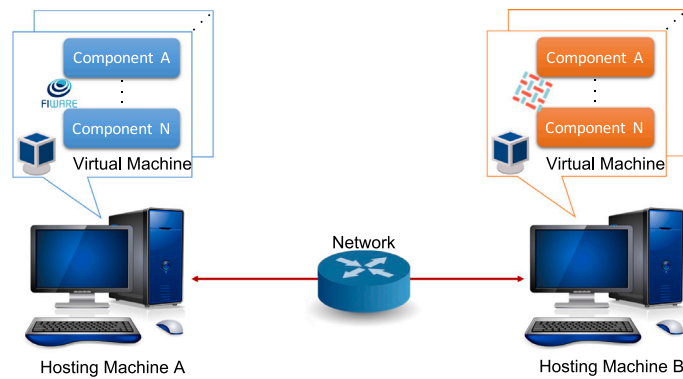


Fig. 12. Testbed deployment.

#### 4. Assessment

The implemented prototype has been deployed as a set of docker images within two personal computers, one hosting the FIWARE components and one our blockchain-empowered elements, both hosted in multiple Virtual Machines (VM) as in Fig. 12. The reference implementation scenario is illustrated in Fig. 13, which represents an architectural view of Fig. 1, by considering the design principles of FIWARE and our proposed solution. Specifically, a set of IoT nodes are deployed across the city, *i.e.*, within the buildings, at the moving vehicles or alongside the roads. Such nodes send a series of monitoring data to the IoT agents of the FIWARE platform within the Fog layer. The Onion Context Broker in FIWARE interconnects the sensing nodes and the upper services implemented as GE running within the cloud. Our testbed neglects the lower layer, and only focuses on simulating the cloud level. Such a level is composed by a set of GEs implementing the logic of the back-end and front-end services and the Service-related operations within the FIWARE platform. Those level is simulated by the VMs and Docker instances running at the Hosting Machine A, where a VM emulated the cloud-hosted services of a given organization. The data of interest for running the Service-related GEs are hosted in the Data Layer of the cloud, thanks to a DB or the set of peers from Hyperledger Fabric. Such a layer is emulated by the Hosting Machine B, where a single VM contains those components deployed by a single organization. Both Db and blockchain is hosted on the same VM so as to have the same running conditions. Within such deployment, the assessment aims at measuring the latency to the main operations implemented by our solution and compare them with the existing solutions available in FIWARE, such as a single database (enabled at a single organization and having the others pointing the unique DB) and the federation of databases (one per each emulated organization), hosted within the Hosting Machine B along with the blockchain deployment.

The first conducted assessment aims at comparing the performance achievable by using the blockchain with the case of using a federation of relational databases by employing a 3PC for guaranteeing the consistency among multiple replicas (three replicas have been used in the experiments running at multiple virtual machines in the hosting machine b). The first experiments consist in sending requests of queries, inserting and updating details related to users in the implemented extension of Keyrock IdM. For the query operation, a mean latency over 20 requests has been equal to about 390 milliseconds when the blockchain is used, while it has been equal to 700 milliseconds by using a federated set of databases. For the insert/update operations, we have respectively measured 3160 milliseconds and 2870 milliseconds to complete them when blockchain is used, while 50 milliseconds and 30 milliseconds with the use of a federated database system. This shows that the blockchain is more advantageous when querying, rather than data management, in which the federated database system is extremely faster as the distributed consensus is not needed. Such a behavior has been seen also at the evaluation of the performance of the blockchain-based access control has been tested against the centralized solution represented by MongoDB. When a peer of Fabric is used as PRP, insert/deleting a policy and retrieving one take about 3945 and 758 milliseconds; while when the MongoDB is used as PRP, the update and query latency in average are equal to 50 and 330 milliseconds, respectively. These higher latency are caused by the consensus algorithm used by Fabric and the securing of the communications by using TLS. In the case of a smart city platform, inserting new users or updating their information is extremely rare, while the query is performed more frequently, making the blockchain solution a promising one.

A second kind of evaluation of the proposed solution is based on the impact on the proposed solution of the number of nodes within a given organization acting as peer for the blockchain. The results shown in Fig. 14 shows that the time needed to obtain a result from the querying of the blockchain reduces considerably with the increase of nodes in each organization. In fact, having more nodes allows parallel chaincode execution, and this positively affects the execution time. Similarly, the throughput, measured as the number of complete transactions in one second, increases due to the positive effect of increasing the number of nodes. Moreover, within the context of Hyperledger Fabric v1.4, the platform provides not only the Kafka-based consensus but also Raft. The first algorithm in crash fault tolerant (CFT), while the second one is a variant of the Paxos consensus and is able to provide Byzantine Fault Tolerance (BFT). In our experiments, we have used Kafka, but we have re-run them by using also Raft, and we have noticed an increase in terms of latency in Fig. 15 (and the consequent reduction in terms of throughput), meaning that the increased BFT guarantees has an impact on the perceived performance of the solution. Another important parameter impacting the performance of

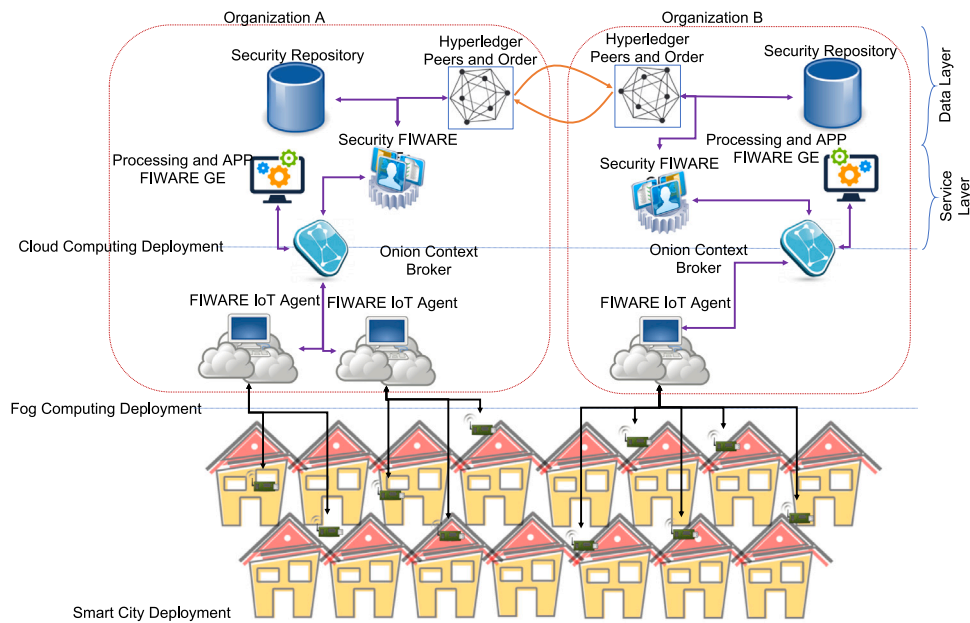


Fig. 13. Implementation Scenario.

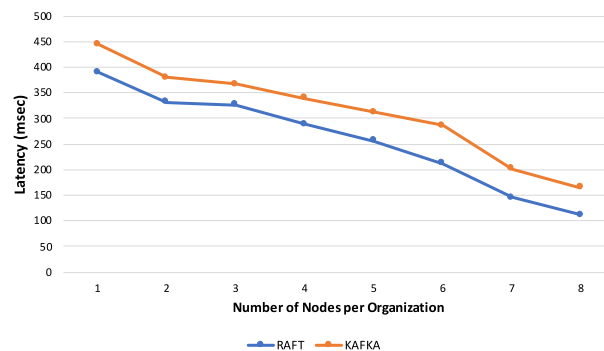


Fig. 14. Latency variations at increasing nodes within an organization and different consensus algorithms.

the blockchain is the block size, influencing the number of transactions can be included in a single block. For this reason, we have assessed the throughput when using Kafka-based consensus and increasing block size, and we have noticed in Fig. 16 a dramatic drop, due to the fact that bigger blocks takes more time to be agreed by the consensus as the higher number of chaincode to be executed, and the additional communication latency caused by larger messages peers need to exchange.

## 5. Conclusions

Within the context of large scale smart cities, the supporting infrastructure is not managed by a single organization. Still, it is realized by integrating and federating multiple infrastructures, each owned by a given company. There is no single root of trust in this multi-tenant model, as each company may have its own access control model and storage. The typical platforms for smart cities, such as FIWARE, are equipped with a centralized approach for identity and access control management, which is not suitable within the considered ecosystem. Therefore, a more distributed approach is needed so that each organization keeps its repository, which is federated with the ones of the others by leveraging on the blockchain technology. This work has proposed the design and realization of the blockchain-based solution for authentication and authorization in smart cities and has integrated it within the context of FIWARE. An assessment of such a solution compared to the existing centralized and existing solutions has been proposed. Our solution can be easily used in other platforms and contexts, as it is generic and each to be coupled to any possible platform due to its intrinsic decoupling. In fact, as a future work, we plan to use it within the context of the IoT to augment our solution presented in Cinque et al. (2020), Esposito, Tamburis, Su, and Choi (2020), and implementing the dynamic access control model described in Esposito (2018), where identity and security policies provided by the solution described in this work represent the first



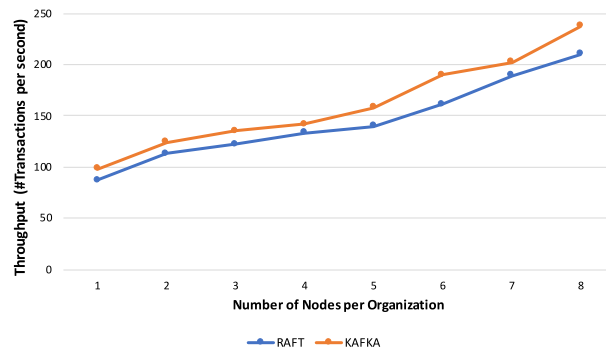


Fig. 15. Throughput variations at increasing nodes within an organization and different consensus algorithms.

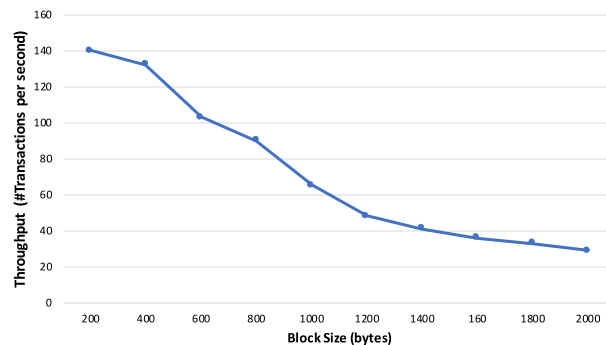


Fig. 16. Throughput variations at increasing block size.

stage of decision, followed by a second that uses the trust estimation. This last step is needed to make the access decision depending on the observed behavior of an entity, rather than its possessed claims/attributes and in the first step. This allows to detect possible malicious behaviors and exclude from the infrastructure potentially compromised components/users.

Last, our work has not addressed the problem of checking the correctness of the stored policies, and/or to detect possible conflicts among stored policies. In fact, it may be possible that policies submitted by two organizations may oppose each other, or may even have semantic heterogeneity or be compliant to different policy models. Therefore, we plan to work on policy checkers to be used to validate the submitted policies before being agreed by the used consensus algorithm in the blockchain, and the possibility of letting heterogeneous policies coexists within the blockchain.

### CRedit authorship contribution statement

**Christian Esposito:** Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing. **Massimo Ficco:** Conceptualization, Methodology. **Brij Bhooshan Gupta:** Software, Writing - review & editing.

### References

- Ahuja, S. P., & Wheeler, N. (2020). Architecture of fog-enabled and cloud-enhanced Internet of Things applications. *International Journal of Cloud Applications and Computing (IJCAC)*, 10(1), 1–10.
- Al-Sharif, Z. A., Al-Saleh, M., Alawneh, L. M., Jararweh, Y. I., & Gupta, B. B. (2020). Live forensics of software attacks on cyber-physical systems. *Future Generation Computer Systems*, 108, 1217–1229.
- Alonso, A., Pozo, A., Choque, J., Bueno, G., Salvachúa, J., Diez, L., et al. (2019). An identity framework for providing access to fiware oauth 2.0-based services according to the eidas european regulation. *IEEE Access*, 7, 88435–88449.
- Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., et al. (2018). Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*.
- Baniata, H., Anagreh, A., & Kertesz, A. (2021). PF-BTS: A privacy-aware fog-enhanced blockchain-assisted task scheduling. *Information Processing & Management*, 58(1), Article 102393.
- Bauer, C. (2017). Openremote v2 documentation.
- Bellamy-McIntyre, J., Luterroth, C., & Weber, G. (2011). Openid and the enterprise: A model-based analysis of single sign-on authentication. In *2011 IEEE 15th international enterprise distributed object computing conference* (pp. 129–138). IEEE.
- Berdik, D., Otoum, S., Schmidt, N., Porter, D., & Jararweh, Y. (2021). A survey on blockchain for information systems management and security. *Information Processing & Management*, 58(1), Article 102397.

- Bertin, E., Hussein, D., Sengul, C., & Frey, V. (2019). Access control in the internet of things: a survey of existing approaches and open research questions. *Annals of Telecommunications*, 74(7–8), 375–388.
- Bhushan, K., & Gupta, B. B. (2019). Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment. *Journal of Ambient Intelligence and Humanized Computing*, 10(5), 1985–1997.
- Biswas, K., & Muthukumarasamy, V. (2016). Securing smart cities using blockchain technology. In *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)* (pp. 1392–1393).
- Borowsky, E., & Gafni, E. (1993). Generalized flip impossibility result for t-resilient asynchronous computations. In *Proceedings of the twenty-fifth annual ACM symposium on theory of computing* (pp. 91–100).
- Bruno, D., Distefano, S., Giacobbe, M., Minnolo, A. L., Longo, F., Merlino, G., et al. (2019). An iot service ecosystem for smart cities: The # smartme project. *Internet of Things*, 5, 12–33.
- Carnevale, L., Celesti, A., Di Pietro, M., & Galletta, A. (2018). How to conceive future mobility services in smart cities according to the fiware frontiercities experience. *IEEE Cloud Computing*, 5(5), 25–36.
- Chen, Q., Srivastava, G., Parizi, R. M., Aloqaily, M., & Al Ridhawi, I. (2021). An incentive-aware blockchain-based solution for internet of fake media things. *Information Processing & Management*, 57(6), Article 102370.
- Cinque, M., Esposito, C., Russo, S., & Tamburis, O. (2020). Blockchain-empowered decentralised trust management for the internet of vehicles security. *Computers and Electrical Engineering*, 86, Article 106722.
- Dabholkar, A., & Saraswat, V. (2019). Ripping the fabric: Attacks and mitigations on hyperledger fabric. In *Applications and techniques in information security* (pp. 300–311). Singapore: Springer Singapore.
- Del Esposte, A. M., Kon, F., Costa, F. M., & Lago, N. Interscity: A scalable microservice-based open source platform for smart cities. In *SMARTGREENS*, vol. 1 (pp. 35–46).
- Du, R., Santi, P., Xiao, M., Vasilakos, A. V., & Fischione, C. (2019). The sensible city: A survey on the deployment and management for smart city monitoring. *IEEE Communications Surveys Tutorials*, 21(2), 1533–1560.
- Enterprise, K. (2016). Kaa documentation.
- Esposito, C. (2018). Interoperable, dynamic and privacy-preserving access control for cloud data storage when integrating heterogeneous organizations. *Journal of Network and Computer Applications*, 108, 124–136.
- Esposito, C., Castiglione, A., Palmieri, F., & Ficco, M. (2017). Improving the gossiping effectiveness with distributed strategic learning (invited paper). *Future Generation Computer Systems*, 71, 221–233.
- Esposito, C., Castiglione, A., Palmieri, F., Ficco, M., Dobre, C., Iordache, G., et al. (2018). Event-based sensor data exchange and fusion in the internet of things environments. *Journal of Parallel and Distributed Computing*, 118(part 2), 328–343.
- Esposito, C., Tamburis, O., Su, X., & Choi, C. (2020). Robust decentralised trust management for the internet of things by using game theory. *Information Processing & Management*, 57(6), Article 102308.
- Ficco, M., Esposito, C., Xiang, Y., & Palmieri, F. (2017). Pseudo-dynamic testing of realistic edge-fog cloud ecosystems. *IEEE Communications Magazine*, 55(11), 98–104.
- Gilbert, S., & Lynch, N. (2012). Perspectives on the cap theorem. *Computer*, 45(2), 30–36.
- Gracia, L. (2018). Sofia4cities documentation.
- Hammer-Lahav, D., & Hardt, D. (2011). *The oauth2. 0 authorization protocol: Technical report*, IETF Internet Draft.
- Hammi, M. T., Hammi, B., Bellot, P., & Serhrouchni, A. (2018). Bubbles of trust: A decentralized blockchain-based authentication system for iot. *Computers & Security*, 78, 126–142.
- Hardt, D., et al. (2012). *The oauth 2.0 authorization framework: Tech. rep. RFC 6749*.
- Khare, A., Merlino, G., Longo, F., Puliafito, A., & Vyas, O. P. (2020). Design of a trustless smart city system: The #smartme experiment. *Internet of Things*, 10, Article 100126.
- Li, D., Deng, L., Gupta, B. B., Wang, H., & Choi, C. (2019). A novel CNN based security guaranteed image watermarking generation scenario for smart city applications. *Information Sciences*, 479, 432–447.
- Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q. (2020). A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107, 841–853.
- Li, J., Wu, J., Jiang, G., & Srikanthan, T. (2021). Blockchain-based public auditing for big data in cloud storage. *Information Processing & Management*, 57(6), Article 102382.
- Liu, Y., Zhao, Z., Guo, G., Wang, X., Tan, Z., & Wang, S. (2017). An identity management system based on blockchain. In *2017 15th Annual Conference on Privacy, Security and Trust (PST)* (pp. 44–4409). IEEE.
- Lockhart, H., & Campbell, B. (2008). Security assertion markup language (saml) v2.0 technical overview. *OASIS Committee Draft*, 2, 94–106.
- McClellan, S. (2019). *Smart cities in application: Healthcare, policy, and innovation*. Springer.
- Miyata, T., Koga, Y., Madsen, P., Adachi, S.-I., Tsuchiya, Y., Sakamoto, Y., et al. (2006). A survey on identity management protocols and standards. *IEEE Transactions on Information and Systems*, 89(1), 112–123.
- Mohamed, N., Al-Jaroodi, J., Jawhar, I., Lazarova-Molnar, S., & Mahmoud, S. (2017). Smartcityware: A service-oriented middleware for cloud and fog enabled smart city services. *IEEE Access*, 5, 17576–17588.
- Mohan, C., & Lindsay, B. (1985). Efficient commit protocols for the tree of processes model of distributed transactions. *Operating Systems Review*, 19(2), 40–52.
- Monrat, A. A., Schelén, O., & Andersson, K. (2019). A survey of blockchain from the perspectives of applications, challenges, and opportunities. *IEEE Access*, 7, 117134–117151.
- Moskvitch, K. (2016). Barcelona: the world's smart city? *Engineering & Technology*, 11(5), 48–51.
- Nguyen, G.-T., & Kim, K. (2018). A survey about consensus algorithms used in blockchain. *Journal of Information Processing Systems*, 14(1), 101–128.
- Novo, O. (2018). Blockchain meets iot: An architecture for scalable access management in iot. *IEEE Internet of Things Journal*, 5(2), 1184–1195.
- Oham, C., Michelin, R. A., Jurdak, R., Kanhere, S. S., & Jhaab, S. (2021). B-FERL: Blockchain based framework for securing smart vehicles. *Information Processing & Management*, 58(1), Article 102426.
- Ouaddah, A., Abou Elkalam, A., & Ait Ouahman, A. (2016). Fairaccess: a new blockchain-based access control framework for the internet of things. *Security and Communication Networks*, 9(18), 5943–5964.
- Paci, F., Squicciarini, A., & Zannone, N. (2018). Survey on access control for community-centered collaborative systems. *ACM Computing Surveys*, 51(1), 1–38.
- Perera, C., Zaslavsky, A., Christen, P., Compton, M., & Georgakopoulos, D. (2013). Context-aware sensor search, selection and ranking model for internet of things middleware. In *2013 IEEE 14th international conference on mobile data management*, vol. 1 (pp. 314–322). IEEE.
- Petrolo, R., Loscri, V., & Mitton, N. (2017). Towards a smart city based on cloud of things, a survey on the smart city vision and paradigms. *Transactions on Emerging Telecommunications Technologies*, 28(1), Article e2931.
- Presser, M., Barnaghi, P. M., Eurich, M., & Villalonga, C. (2009). The sensei project: integrating the physical world with the digital world of the network of the future. *IEEE Communications Magazine*, 47(4), 1–4.
- Putz, B., Dietz, M., Empl, P., & Pernul, G. (2021). EtherTwin: Blockchain-based secure digital twin information management. *Information Processing & Management*, 58(1), Article 102425.

- Riabi, I., Ayed, H. K. B., & Saidane, L. A. (2019). A survey on blockchain based access control for internet of things. In *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)* (pp. 502–507). IEEE.
- Sakimura, N., Bradley, J., Jones, M., De Medeiros, B., & Mortimore, C. (2014). *Openid connect core 1.0* (p. S3). The OpenID Foundation.
- Salman, T., Zolanvari, M., Erbad, A., Jain, R., & Samaka, M. (2019). Security services using blockchains: A state of the art survey. *IEEE Communications Surveys & Tutorials*, 21(1), 858–880.
- Sanchez, L., Muñoz, L., Galache, J. A., Sotres, P., Santana, J. R., Gutierrez, V., et al. (2014). Smartsantander: Iot experimentation over a smart city testbed. *Computer Networks*, 61, 217–238.
- Shao, W., Jia, C., Xu, Y., Qiu, K., Gao, Y., & He, Y. (2020). Attrichain: Decentralized traceable anonymous identities in privacy-preserving permissioned blockchain. *Computers & Security*, Article 102069.
- Standard, O. (2005). Extensible access control markup language (xacml) version 3.0.
- Tan, R., Chirkova, R., Gadepally, V., & Mattson, T. G. (2017). Enabling query processing across heterogeneous data models: A survey. In *2017 IEEE international conference on Big Data (Big Data)* (pp. 3211–3220). IEEE.
- Tewari, A., & Gupta, B. B. (2020). Security, privacy and trust of different layers in Internet-of-Things (IoT's) framework. *Future Generation Computer Systems*, 108, 909–920.
- Tourani, R., Misra, S., Mick, T., & Panwar, G. (2017). Security, privacy, and access control in information-centric networking: A survey. *IEEE Communications Surveys & Tutorials*, 20(1), 566–600.
- Wang, X., Zha, X., Ni, W., Liu, R. P., Guo, Y. J., Niu, X., et al. (2019). Survey on blockchain for internet of things. *Computer Communications*, 136, 10–29.
- Wang, S., Zhang, Y., & Zhang, Y. (2018). A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access*, 6, 38437–38450.
- Xiao, Y., Zhang, N., Lou, W., & Hou, Y. T. (2020). A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22(2), 1432–1465.
- Xie, J., Tang, H., Huang, T., Yu, F. R., Xie, R., Liu, J., et al. (2019). A survey of blockchain technology applied to smart cities: Research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(3), 2794–2830.
- Zahed Benisi, N., Aminian, M., & Javadi, B. (2020). Blockchain-based decentralized storage networks: A survey. *Journal of Network and Computer Applications*, 162, Article 102656.
- Zhang, P., Liu, J. K., Yu, F. R., Sookhak, M., Au, M. H., et al. (2018). A survey on access control in fog computing. *IEEE Communications Magazine*, 56(2), 144–149.
- Zhao, Q., Chen, S., Liu, Z., Baker, T., & Zhang, Y. (2021). Blockchain-based privacy-preserving remote data integrity checking scheme for IoT information systems. *Information Processing & Management*, 57(6), Article 102355.
- Zhu, X., & Badr, Y. (2018). Identity management systems for the internet of things: a survey towards blockchain solutions. *Sensors*, 18(12), 4215.