IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 3, 2020, accepted May 8, 2020, date of publication May 27, 2020, date of current version June 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2998012

A Framework Based on Distributed Ledger Technologies for Data Management and Services in Intelligent Transportation Systems

MIRKO ZICHICHI[1,2,3], STEFANO FERRETTI[4], (Member, IEEE), AND GABRIELE D'ANGELO[2], (Member, IEEE)

[1]Law, Science, and Technology Joint Doctorate—Rights of Internet of Everything
[2]Department of Computer Science and Engineering, University of Bologna, 40125 Bologna, Italy
[3]Ontology Engineering Group, Universidad Politécnica de Madrid, 28660 Madrid, Spain
[4]Department of Pure and Applied Sciences, University of Urbino "Carlo Bo," 61029 Urbino, Italy

Corresponding author: Mirko Zichichi (mirko.zichichi@upm.es)

This work was received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie ITN EJD grant agreement No 814177 Law, Science, and Technology Joint Doctorate—Rights of Internet of Everything.

ABSTRACT Data are becoming the cornerstone of many businesses and entire systems infrastructure. Intelligent Transportation Systems (ITS) are no different. The ability of intelligent vehicles and devices to acquire and share environmental measurements in the form of data is leading to the creation of smart services for the benefit of individuals. In this paper, we present a system architecture to promote the development of ITS using distributed ledgers and related technologies. Thanks to these, it becomes possible to create, store and share data generated by users through the sensors on their devices or vehicles, while on the move. We propose an architecture based on Distributed Ledger Technologies (DLTs) to offer features such as immutability, traceability and verifiability of data. IOTA, a promising DLT for IoT, is used together with Decentralized File Storages (DFSes) to store and certify data (and their related metadata) coming from vehicles or by the users' devices themselves (smartphones). Ethereum is then exploited as the smart contract platform that coordinates the data sharing through access control mechanisms. Privacy guarantees are provided by the usage of distributed key management systems and Zero Knowledge Proof. We provide experimental results of a testbed based on real traces, in order to understand if DLT and DFS technologies are ready to support complex services, such as those that pertain to ITS. Results clearly show that, while the viability of the proposal cannot be rejected, further work is needed on the responsiveness of DLT infrastructures.

INDEX TERMS Intelligent transportation systems, distributed ledger technologies, blockchain, smart contracts, decentralized file storage, sensing as a service.

I. INTRODUCTION

The future of Intelligent Transportation Systems (ITS) will be based on the ability of vehicles to sense, store and exchange big data. Vehicles will be more and more equipped with sensors that track information about the vehicle internals, as well as information about the surrounding environment and road conditions. Moreover, ubiquitous connectivity allows individuals to post crowdsensed information through their smartphones and mobile devices. This makes them become an active part of ITS. Such crowd-sensed information is essential for building sophisticated smart services that aim at improving traffic management, transportation efficiency and safety, raising awareness about the environment, and thus improving the liveability and health status of the community of a given territory. We thus envisage that vehicles and their users will be able to record data, store them in some data boxes, and communicate with other vehicles or users as well.

There are numerous examples of Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) based applications, such as notification services [1], [2], as well as standards for communication messages, e.g., ETSI Cooperative Awareness

The associate editor coordinating the review of this manuscript and approving it for publication was Yanli Xu.

Messages (CAM) [3]. In these contexts, one of the main issues is the unreliability of the exchanged information. This problem is typically due to the physical errors of the sensors, malfunctions, poor network and GPS coverage. Such noised data lead to inaccurate information. Another problem is due to the fact that some users might be interested in deliberately transferring forged information. Examples are insurance frauds, as well as free-riders that decide to share false data, randomly generated without using their sensors, in order to gain some revenues/credits for such fake data sharing. Thus, a main goal to pursue is the identification of strategies for the generation and distribution of secure and trustful crowd-sourced information.

Meanwhile, different initiatives are growing with the aim of enhancing mobility services (e.g. MOBI [4]). Among these, some focus on frameworks for sharing mobility data provided by users [5]. These solutions typically resort to Distributed Ledger Technologies (DLTs) and other related mechanisms. The rationale behind this choice is the need for trustful data trading and sharing, allowing anyone to verify the authenticity and the immutability of information (see Figure 1).

DLTs can be seen as the evolution of the well known blockchain, which gathered momentum after the sudden increase of Bitcoin demand [6], and after the introduction of the Ethereum decentralized platform, featuring smart contract functionalities [7], [8]. The appeal of these technologies is mainly due to the fact that they avoid the possibility of downtime, censorship, fraud or third party interference. This led additional interest on researchers that then introduced more advanced DLTs, distributed data storage systems, as well as security and cryptography schemes for verifiable encryption, zero-knowledge proofs, and data secrecy [9]–[11]. These technologies can be proficiently employed in several application domains [12]–[15], as well as ITS scenarios [1], [16]–[18].

The "intelligent" word, in ITS, means that data generated by users' smartphones, vehicles' sensors or IoT devices, are transformed into new meaningful information useful for individuals and the ecosystem itself. Hence, one of the main issues is to provide means to easily publish data, while granting compliance with the (related) individuals' privacy preferences and regulations, e.g. the EU GDPR [19]. We think that the use of non-centralized technologies can inhibit the tendency to transfer data in the hands of few entities, that often operate without transparency. Rather, decentralized architectures might promote individuals' data sovereignty and the possibility of the creation of fair data marketplaces. Indeed, any individual in the ITS can share data coming from his vehicle or his personal device (e.g. smartphone) becoming a data provider. Subsequently, anyone else can access these data, with permissions granted following an agreement and thus becoming a data consumer (see Figure 2).

In this paper, we describe a decentralized software architecture that fulfils the mentioned "desiderata". In our architecture, data sharing services are defined to let users and IoT

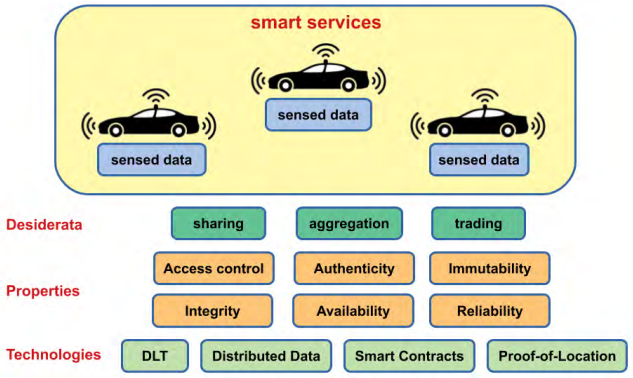

FIGURE 1. Transportation systems: desiderata, features and technologies.

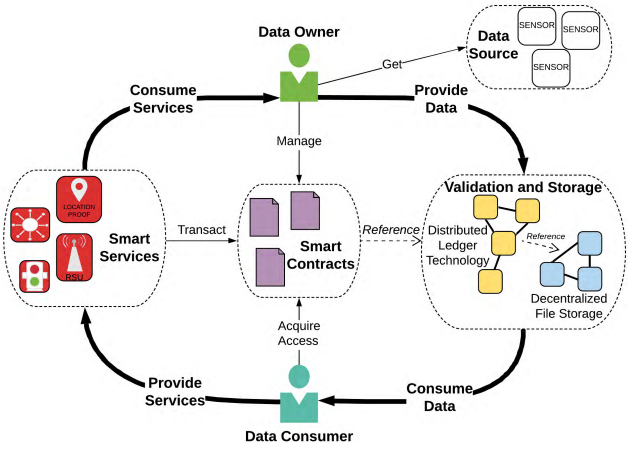

FIGURE 2. Data life cycle: data owners get data from their sensors to provide these to data consumers through the use of DLTs and smart contracts; consumers are then able to generate new data or to provide smart services accessible by individuals such as data owners.

sensors to share their data. These services permit to define how data can be shared, but also how data are acquired. The proposed infrastructure is based on DLTs, in combination with other technologies for Decentralized File Storage (DFS), i.e. IPFS [9] and Sia [20]. Access to crowd-sensed data is regulated through smart contracts, that implement a control list and provide access only to authorized users.

We developed a software architecture where vehicles generate data (e.g. obtained through sensors) that are stored and shared thanks to the combined use of IOTA [21] and a DFS. Ethereum smart contracts provide the coordination among entities. Thus, smart contracts allow gaining access to specific data of interest, once the user has the authorization to access the data (e.g. by paying for such access authorization). Authorization for data access is achieved through a distributed key management system. Moreover, Zero Knowledge Proof is employed as the scheme to offer privacy while providing proof-of-location guarantees. The developed proof of concept implementation demonstrates the viability of the devised solution and it is freely available on GitHub [22], [23].

After defining the architecture, it was important to assess its performance as a whole and for each building block. For this reason, we conducted a detailed performance evaluation, focusing explicitly on the use of the distributed technologies for data sharing. Indeed, in the proposed architecture the data publication is one of the most time-critical parts of the data life cycle. On this aspect, is it worth noticing that the DLT technologies are in its infancy and it is necessary to investigate the viability of their adoption and their scalability. Furthermore, several approaches can be pursued for storing data in DFS.

To sum up, the main contribution of this paper is threefold:

- We propose a layered, decentralized software architecture for the development of novel services for ITS. To this aim, the architecture makes use of novel decentralized technologies, such as DLT, DFS, smart contracts, proof-of-location and sophisticated authorization schemes. We claim that this choice promotes individuals' data sovereignty.
- We describe the implementation of a prototype system architecture. In particular, we used IOTA and its MAM channels as the main DLT to store transactions in the ledger; Ethereum as the platform to implement and execute smart contracts that govern data access; IPFS and Sia as the DFS to store large data files; secret sharing as the main scheme to implement the authorization service; we employ proof-of-location as an exemplar of certificate that provides trust of data; we developed a payment system that allows trading data for fungible (ERC-20) tokens in novel ITS smart services, based on the μRaiden framework [22], [23]. To the best of our knowledge, this is the first work that proposes an integrated use of all these technologies.
- We provide experimental results of a real testbed evaluation of the most critical aspects of the implemented software architecture, so as to understand the viability on the use of the current available technologies for ITS. In particular, through a trace-driven simulation, in our tests we generated a data traffic mimicking a fleet of buses, traveling in Rio de Janerio, that periodically sense data and send them to the system infrastructure. Such data traffic was submitted to the IOTA DLT and the employed DFS, under different real setups. Outcomes demonstrate that, while the system architecture and its components are very promising, work has to be made in order to make decentralized technologies more responsive and scalable.

The remainder of this paper is organized as follows. Section II introduces some background and related work. Section III presents the proposed distributed software architecture. Section IV discusses a use case based on the development of a data marketplace. Section V describes the design of the experimental evaluation we conducted and the obtained results. Section VI provides a discussion on the obtained results. Finally, Section VII provides some concluding remarks.

II. BACKGROUND AND RELATED WORK

In this section, we review the main background concepts and technologies that are needed to present the proposed decentralized software architecture. We also review some of the main related works in the literature.

A. BACKGROUND

The key building blocks at the basis of our system architecture are VANETs, DLTs, DFSes and encryption techniques. In the rest of this subsection, we provide some related background for each of these main topics.

1) VANETs

Vehicular Ad-hoc NETworks (VANETs) are a decentralized type of networks among vehicles, usually considered as the reference use case for ITS. VANETs allow vehicles to form a peer-to-peer substrate to share information and for the creation of smart mobility applications [17], [24]. The term VANET was originally introduced as the vehicular-based specification of general Mobile Ad-Hoc Networks (MANETs). Thus, the focus was in particular, on the ad-hoc construction of an overlay network among cars. However, today VANET is mostly considered as a synonym of the more generic term "vehicular network".

VANETs are thus networks of vehicles, representing network nodes. VANETs exploit wireless communications, forming a landscape where vehicles can communicate between each other (V2V), with some fixed on-road equipment (V2X) or with the transport infrastructure (V2I).

In order to participate to the VANET, each vehicle can be equipped with two components: an Application Unit (AU) and an On Board Unit (OBU). The AU stores an application software, that uses communication and sensing services provided by the OBU. Vehicles can communicate to the Internet by means of a Road Side Unit (RSU), that can be seen as a gateway to the wired network infrastructure, but they can also offer different kinds of services. Indeed, RSUs typically host applications that provide services, while OBUs and AUs are devices that use the provided services.

2) DISTRIBUTED LEDGER TECHNOLOGIES

A Distributed Ledger Technology (DLT) is a technical implementation of a data ledger, thought with the aim to move trust from a human intermediary, that manages a transaction between two parties, to a protocol that allows the two parties to transact directly, i.e. without the need of a third party [17]. The ledger ensures immutable persistence of data, thus providing untampered data to applications when it is necessary. For this reason, DLTs represent an appealing technology for the development of trustful and reliable ITS services [17], [18].

There are different implementations of DLTs, each one with its pros and cons. First, they can be subdivided in two main categories: i) permissionless, i.e. anyone can have access to the peers' network and to participate in the

consensus mechanism, and ii) permissioned, i.e. access are ruled through a hierarchy of participants. Hyperledger Fabric is an example of permissioned DLT [25]. The permissioned approach can be very convenient in many situations, and this solution eases the composition of a software architecture. However, in order to use a permissioned DLT, a consortium of trusted entities is needed. These entities usually act as certificate authorities that release public and private keys to access the ledger [26]. Obviously, such a solution requires to trust such consortium [27]. In contrast, we think that a permissionless approach, such as Ethereum and IOTA DLTs, is more suitable to let every individual share his data and enjoy services. For this reason, in this work we focus on permissionless DLTs, and we propose a solution that avoids trusting a centralized governance.

Another main distinction among DLTs lies on their possible support to smart contracts. This feature is quite often in contrast with other key features, related to the level of scalability and responsiveness. For example, Ethereum [7] provides a distributed virtual machine able to process any kind of computation through smart contracts. However, it is well known such blockchain has some scalability issues [28]. Conversely, the IOTA ledger [21] is thought to provide better scalability, but it currently does not support smart contracts.

Trying to sum up, if one looks at the solutions that are available at the time of writing, there is no single, operative and fully fledged solution that is able to cover all the features that are needed (and reported in Figure 1). Thus, we claim that in order to build a sophisticated software architecture, that is able to act as the middleware for secure and certified ITS applications, multiple DLTs need be utilized and combined together, so as to take the best of multiple worlds. This is the philosophy we followed in our approach.

a: SMART CONTRACTS AND PAYMENT CHANNELS

Smart contracts provide a new paradigm where an immutable set of instructions is deterministically executed during a transaction between two parts. Without the presence of a third party, the execution of a smart contract is performed in such a way that a contract issuer can always be sure that the behavior he implemented is observed. In the case of Ethereum, every process is completely traced and permanently stored in the blockchain. Moreover, the smart-contract computation is executed by all network participants.

Since transactions in smart contracts and blockchains can be expensive in terms of fees and latencies, payment channels have been introduced to implement rapid micropayments. We can distinguish between on-chain and off-chain payments. On-chain payments can be made, for instance, through smart contracts implementing ERC20-like tokens and cryptocurrencies [29]. Off-chain payments, on the other hand, allow users to perform multiple payments, with the advantage that only the first and the last payment transactions are stored into the ledger. To guarantee the payment security, off-chain payments are regulated through smart contracts (or scripts in the case of Bitcoin), that manage the so called state channel. The most well known examples of off-chain payment schemes are the Bitcoin's Lightning Network [30] and the Ethereum's Raiden Network [31]. The main advantage of this approach is the limited amount of fees that are usually paid, since the majority of the transactions are not registered in the blockchain. Another effect of this approach is the faster confirmation of transactions.

More in detail, the typical approach followed to implement micropayments has three simple steps: i) a user (sender) that is willing to pay a provider (receiver) for a service *opens* a channel through a dedicated smart contract, depositing the amount of currency that can be possibly transferred in that channel; ii) both sender and receiver communicate using an off-chain channel, in order to *update* the balance; iii) sender and receiver can *close* the channel by invoking the smart contract and submitting the last agreed balance message and the other party signature. Then, the balance and the remaining deposit will be moved to their accounts.

b: IOTA

IOTA is a permissionless DLT that allows hosts in a network to transfer immutable data among each other. It is specifically designed for the IoT industry. The ledger used in IOTA is not structured as a blockchain but as a Direct Acyclical Graph (DAG) called the Tangle [21]. In the IOTA DAG, the graph vertices represent transactions and edges represent approvals. When a new transaction is issued, it must approve the two previous transactions and the result is represented by means of directed edges. The process of attaching a transaction to the Tangle includes two sub-processes:

- **Tips selection** - To attach a transaction to the Tangle, it is mandatory to reference two previous transactions called tips, i.e. transactions that do not have a successor yet. These tips are provided by the IOTA full node that stores the full graph and are selected using a Markov Chain Monte Carlo algorithm.
- **Proof-of-Work (PoW)** - This step consists in validating a transaction by performing a PoW i.e. a crypto-puzzle computation. Such puzzle consists in making some (costly and time-consuming) computation, in order to obtain a piece of data which satisfies certain requirements. A main characteristics is that the result of the puzzle is difficult to find, but it is easy to verify, once revealed. The purpose of PoW is to deter denial of service attacks and other service abuses (e.g. the 50% + 1 attack).

The validation approach is thought to address two major issues of traditional blockchain-based DLTs, i.e. latency and fees. IOTA has been designed to offer fast validation, and no fees are required to add a transaction to the Tangle [10]. The IOTA design makes it an appealing choice, that in our opinion deserves experimental studies, so as to understand if it can support smart services built through crowd-sourced data. However, we point out that currently a ''coordinator node'' is used in the system. Such central node, maintained

by the IOTA foundation, has the task to perform a periodic checkpointing of the ledger, i.e. releasing milestone transactions that confirm that all the previous ones are valid. The rationale is that the coordinator should sustain possible large-scale security attacks, during the first period of the deployment of this DLT. Indeed, as shown after the ''Trinity wallet attack'' at the start of 2020 [32], the IOTA foundation has the complete ability to shut down the network, simply by halting the coordinator. Clearly enough, this is a big issue that needs to be addressed, for an effective IOTA deployment and in order to foresee a widespread adoption of this technology [33].

An important feature offered by IOTA is the Masked Authenticated Messaging (MAM). MAM is a data communication protocol built upon the Tangle, which adds the functionality to emit and access an encrypted data stream over the Tangle. Data streams assume the form of channels, i.e. a linked list of ordered transactions. Once a channel is created, only the channel owner can publish encrypted messages on it. Users that possess the MAM channel encryption key (or set of keys, since each message can be encrypted using a different key) are enabled to decode the message. Messages are pushed on the channel in chronological order, and each message has a link to the next message to be created. Thus, once a user gains access to the MAM channel, he is enabled to see data from that moment on, whilst he cannot look back through the history of the channel before his entrance. In other words, MAM enables users to subscribe and follow a stream of data, generated by some device. Our architecture makes an extensive use of this feature.

3) DECENTRALIZED FILE STORAGE (DFS)

Decentralized file storage is a potential solution for maintaining files in a system without having to rely on a large, centralized silos that may not completely assure privacy and freedom of information. Such technologies are fundamental for DLTs, since these can be used when the ledger and the consensus mechanism disincentives data storing.

a: IPFS

The InterPlanetary File System (IPFS) [9] is a protocol that builds a distributed file system over a peer-to-peer network. IPFS creates a resilient file storage and sharing system, with no single point of failure and in which the nodes do not need to trust each other. This technology is useful to store data that is not convenient to put on DLTs.

Files published in the IPFS network take the form of IPFS objects. In order to retrieve an object, only the file digest is needed, i.e. the result of an hash function applied on the file. Put in other words, the file digest is the identifier of the IPFS object. Users that want to locate that object use this identifier as an handle.

It is important to mention that, when there is no incentive to maintain files, IPFS does not offer guarantees on the persistence of data in the storage system. It stores data as long as some IPFS node maintains a replica of that file. The more the nodes that maintain a copy of a given file, the higher the reliability and the higher the guarantees that the file can be properly retrieved. However, incentivation mechanisms can be employed to obtain that the distributed system permanently stores files, i.e. users can reward nodes that maintain a copy of their data.

b: INCENTIVIZED SOLUTIONS FOR PERSISTENCE

In order to provide incentives to nodes for maintaining data, some DFSes integrate DLTs, bringing together clients' requests with storage nodes' offers. Filecoin is the typical incentive used on top of IPFS [34]. In practice, participants are rewarded (with Filecoin tokens) for serving and hosting content on their storage. Thus, this strategy does not alter the protocol on how nodes exchange data files. Simply, certain IPFS nodes are paid to store and not erase them.

Besides Filecoin, which is the most showcased solution [35], other solutions exist that provide incentives to persistently store data [20], [36], [37]. For instance, Sia [20] is a DFS that, similarly to Filecoin, integrates a blockchain in order to reward hosts for keeping files. It uses File Contracts, i.e. a particular kind of smart contract employed to arrange an agreement between a storage provider and their clients. While interesting, it is worth mentioning that at the time of writing Sia lacks the simplicity and the level of maturity provided by IPFS. Probably for this reason, current solutions in literature are mostly based on IPFS [38]–[40].

4) CRYPTOGRAPHY TECHNIQUES

The distributed technologies, used for providing smart mobility solutions, are based on DLTs and DFSes. According to these schemes, data is usually public and, therefore, can be accessed by any participant. In certain application scenarios, this can represent a issue. To mitigate this possible problem, in this section we mention some sophisticated cryptographic techniques that allow providing privacy to users.

Zero Knowledge Proof [41] is a protocol in which one party (called Prover) can prove to another party (called Verifier) that he knows a value x without giving any information except the fact that he knows x (i.e. without disclosing the x value).

Proxy Re-Encryption (PRE) [42] is a cryptographic primitive, where an untrusted proxy can translate a message m, encrypted with a key k_1, into a cipher text with key k_2, without being able to see the plain text. This is possible using a re-encryption key rk generated by the user who owns k_1.

Secret sharing, firstly proposed by Shamir in 1979 [43], consists in a method used to distribute a secret amongst a group of n participants, where any group of t (with $t \leq n$) or more of them can together reconstruct the secret, but no group fewer than t in number can. Such scheme is called (t, n)-threshold. Multiple servers form an overlay network; all servers store secrets, shares and provide them to data consumer. This improves user's data privacy, since none of the servers can obtain the whole secret used for encryption without the help of other $t - 1$ servers.

B. RELATED WORK

DLTs have been recently adopted in several contexts, such as IoT [44], [45], smart cities [46], [47] and ITS [26], [48]. The main reason is due to their ability to enable public verifiability of digital transactions and data.

In [49], Yuan and Wang conduct a preliminary study of blockchain based ITS, giving the basis for a new ITS-oriented blockchain model. In this work, the focus is on the blockchain potential to help establishing a secured, trusted and decentralized ecosystem. Leiding *et al.* [16] present CHORUS Mobility, a decentralized system that combines VANETs and Ethereum to provide services and enforce rules in ITS [50]. Sharma *et al.* [51] show various use cases in order to validate blockchain based applications for social good.

In environments such as ITS, where data sharing is fundamental, how the data are obtained is a crucial task, that might require a change of paradigms and employed technologies. Chiasserini *et al.* [1] propose an architecture to validate the information contained in CAMs and provide a tamper-evident and distributed data storage, using DLTs and smart contracts.

Various works in the literature share the main goal we pursue in this paper, i.e. the proposal of a blockchain-based architecture for data management [52], [53]. However, the focus was usually limited in certain aspects of the general problem. Focusing on data trading, a basis for a marketplace is needed. Works such as [54], [55] propose the of use a blockchain to trade data in IoT and smart cities scenarios. López and Farooq [56] build a framework on top of Hyperledger Fabric, where participants can share their encrypted smart mobility data. While maintaining similar features, our solution is completely based on permissionless technologies. Zhang *et al.* [57] design a blockchain-based architecture for data sharing, with attribute-based access control. They use Ethereum smart contracts to grant data access, which are integrated with an attribute encryption scheme [58]. In [59], the authors make use of IOTA and micropayments for streaming data with payment processing and auditable records in a IoT scenario.

The use of IOTA in transportation systems is becoming relevant. In fact, it seems that more and more companies are starting to appreciate its potential [60]. Overko *et al.* [61] present a distributed reinforcement learning system based on IOTA to determine an unknown distribution of traffic patterns in a city. The limitations and potential impact of IOTA in ITS are also studied by Bartolomeu *et al.* in [62]. They propose the use of this DLT to improve the security of both in-vehicle and off-vehicle functions.

Finally, there are other related works, whose applications are not related to ITS, but which share some similarities in terms of the exploited underlying technologies [38], [63]. Among them, Wang *et al.* [40] provide a data sharing framework based on attribute-based encryption, where Ethereum smart contracts are combined to IPFS. Hawig *et al.* [39] propose an architecture based on both MAM channels and IPFS for the exchange of blood glucose data. Their aim is to propose a system that provides immutability, interoperability, and GDPR compliance.

To the best of our knowledge, the work we propose in this paper is the first one that combines the usage of both MAM channels and DFSes, such as IPFS or Sia.

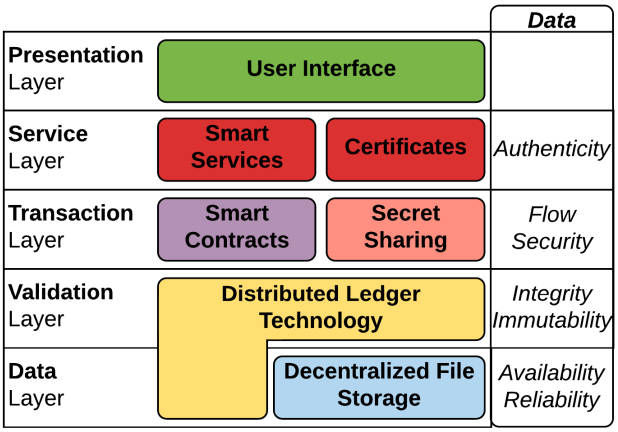

FIGURE 3. System layered architecture.

III. SYSTEM ARCHITECTURE

The system architecture that we propose is based on the aggregation of different distributed technologies that allow collecting and sharing crowd-sensed and user-managed data. Smart services enforce the ability of our architecture to obtain proper, verified data, and to build applications on top of them.

A layered representation of the system architecture is provided in Figure 3. In particular:

- **Data Layer**: the first (lowest) layer comprises those technologies that provide storage and data availability and reliability through replication, i.e. DLTs and DFSes.
- **Validation Layer**: the integrity of data, sensed by users and stored in the data layer, must be guaranteed and verified. To this aim, the validation layer employs DLTs.
- **Transaction Layer**: the use of data is authorized only to entitled users. Access control is performed through smart contracts and secret sharing techniques.
- **Service Layer**: access to shared data allows creating smart services, useful to other users. Needless to say, data consumers must be confident that data are reliable and trustful, e.g. data have been generated in a certain location and measured in proper conditions. Certificates can grant this trust.
- **Presentation Layer**: users can interact with the services in different ways, based on the specific application. This layer is devoted to implement the interfaces between the system architecture and the outside world.

The architecture is fundamentally based on data gathered by sensors placed in vehicles or in users AUs. The user's AU, indeed, represents an ITS node controlled by the user itself, allowing him to share his data and to use smart services: given its main role, it is placed at the center of the diagram

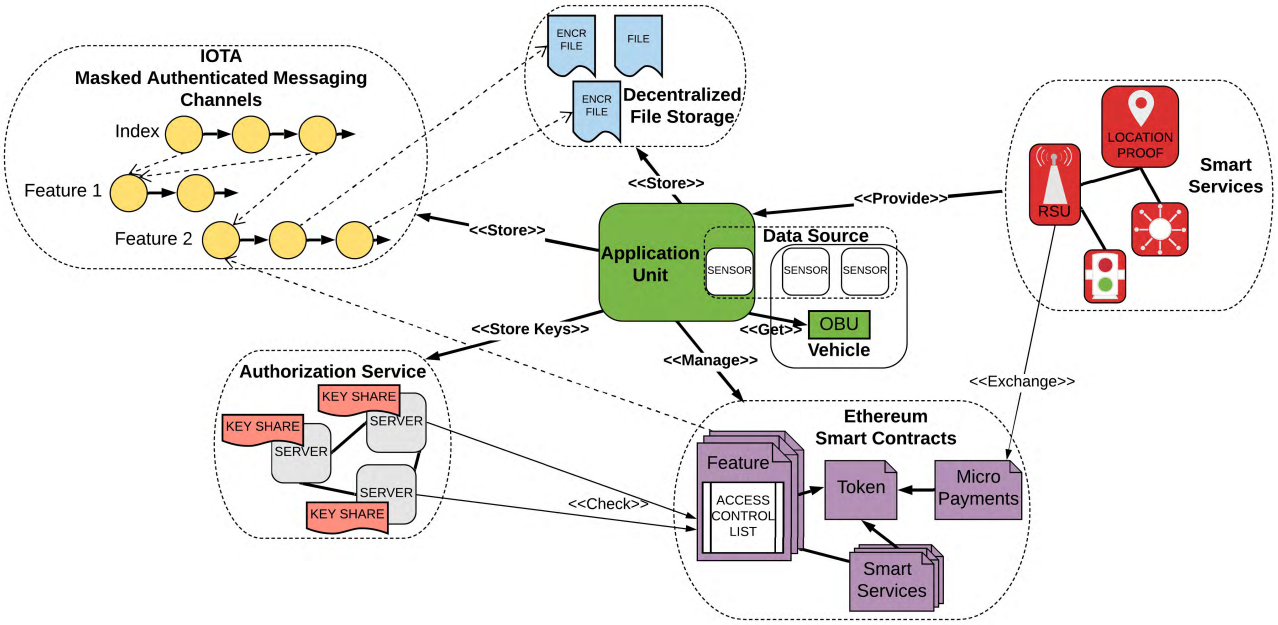

FIGURE 4. System architecture.

in Figure 4. This diagram is intended to give a first glance to the relations in the system and will be further explained in detail in the following.

A. DATA LAYER

Taking Figure 4 as a reference, and in particular its central (green) components, we consider vehicle's internal sensors (controlled through the OBU) and/or user's AU, as the main sources of data sensed in the ITS. Such raw data are then organized and refined to be stored in the distributed system.

Data can be stored in two different technologies: a DFS or a DLT. The discriminating factor for the selection of the specific technology is the data size. Storing data into a DFS usually requires lower latencies with respect to those that can be obtained using DLTs. (In fact, DLTs typically require some time consuming Proof-of-Work.) However, validation is obtained through the publication of the data (or of the data digest) into a DLT (see Validation Layer). Hence, with the aim of fastening the publication process, we adopt the following heuristics:

- Data that do have a large-size, higher than a single sensed value, are stored into a DFS and referenced in the DLT through their ID, e.g. digest (see blue upper components in Figure 4).
- Small size files (whose size is comparable to the size of their digest) are directly stored in the DLTs (yellow components in the figure).

Once a file is published in the DFS, the returned reference can be employed to retrieve it in the network. Taking, for instance, IPFS [9] as the used DFS, such reference is in fact the data digest itself, that it is stored in the DLT (Section III-B). Thus, the piece of data is published as an IPFS object and then (asynchronously) referenced through its hash into a MAM transaction. The digest allows verifying the integrity of the IPFS object. To upload files on IPFS, a node running the IPFS protocol is necessary. Due to the fact that it is (still) not feasible to run an IPFS node on constrained devices (such as smartphones or sensors), other solutions must be explored. For example, in our architecture we assume that an IPFS service provider (e.g. Infura [64]) lets a user permanently store files in the IPFS network, as long as they reach an agreement (e.g. by paying a subscription). An alternative solution to reach such agreement can be automated using a Filecoin smart contract [34].

B. VALIDATION LAYER

One of the aims of the proposed architecture is to give ownership of data to the users that produce them, without having the entire collection of sensed data stored in a centralized data storage service. DLTs allow avoiding all the typical drawbacks of server-based approaches (e.g. censorship, single point of failure), and offer features such as data immutability, verifiability and, most importantly, traceability. During the implementation of the system architecture, we decided to employ IOTA as DLT. At the time, the rationale behind this choice was due to its promises in performances and for the presence of Masked Authenticated Messaging (MAM), a method to handle data that deals with the main requirement of data privacy.

1) IOTA MASKED AUTHENTICATED MESSAGING (MAM)

As already mentioned, in IOTA the Tangle stores immutable information that cannot be censored/removed. The Tangle is a public data ledger, accessible by anyone. Thus, in order to obfuscate data and make them accessible only to authorized entities, MAM channels are used to store encrypted data, providing access only to eligible users.

Algorithm 1 Publish Data for Validation

Data: t data size threshold, *wallet*
Input: p data packet
Input: id_{FC} Feature Channel (FC) id
Result: p published in the FC

```
// wallet is securely accessed from the AU storage
1  k_enc, addr_FC ← wallet(id_FC) // encryption key and address where to store data in the FC
2  if size(p)> t then
3  |   p_enc ← encrypt(p, k_enc)
4  |   digest ← storeOnDFS(p_enc) // returns p_enc SHA256 digest
5  |   p ← digest
6  end
   // start attach to MAM procedure
7  m ← createMAMMessage(p, k_enc)
8  b ← createTangleTXsBundle(addr_FC, m) // bundle composed by m and signature TXs
9  τ1, τ2 ← getTangleTips()
10 b' ← POW(b, τ1, τ2)
11 broadcast(b')
```

Data gathered from sensors are organized in features, i.e. a particular kind of data (e.g. a sensed temperature) or a data point such as geo-location (e.g. the data has been produced in Copacabana, Rio De Janeiro, Brazil) or vehicle's velocity. Two different types of MAM channels are associated to each user, in which data are stored or referenced:

- **Feature Channels** - Each feature has its own MAM channel. A feature channel is, thus, a list of messages containing data or metadata of the same feature, arranged in a chronological order. It can be considered as a continuous log, in which each message has the same structure and contains the data or a reference (i.e. hash pointer to an IPFS object) to these data. Since messages are in chronological order, but with no distinction between sessions (e.g. all data gathered in one particular day), and given the fact that a channel could be terminated in favor of a new one (e.g. in case of credentials loss), a specific kind of channels is needed, in which messages represent a reference to other messages in feature channels. For this reason, we introduce the reference channels.
- **Reference Channels** - These MAM channels are used to reference other channels. There are two types of reference channels, i.e. ''Session'' channels and ''Index'' channels. Each ''Session'' channel records all the sessions related to a given user. For each new session, a novel message is generated and stored in the user's ''Session'' channel. This message contains the reference to all the active ''Feature'' channels of this specific user [17]. The ''Index'' channel is used to maintain the hierarchical structure, needed to reference all the user's MAM channels. When a new channel is created (both ''Session'' or ''Feature''), then a new message is inserted in the ''Index'' channel, containing its reference.

2) PUBLISHING ON IOTA

During the publication of data on IOTA (i.e. validation process, Figure 3), a wallet must hold user's credentials. In our architecture, the AU manages a wallet, which consists in a piece of software that securely handles confidential data. For example, the wallet maintains the private keys used for signing transactions and managing MAM channels. It is implemented as a hierarchical deterministic wallet [65].

The data validation process is shown in Algorithm 1. It mainly consists in retrieving the required keys from the wallet (line 1), generating the appropriate MAM message in the form of a bundle of transactions (TXs, lines 7-8) and executing the process to attach the bundle to the Tangle (usually performed by a IOTA full node, lines 9-11). More specifically, MAM channel messages' payload contains sensor measurements, timestamps and it is represented using standard notations (e.g. JSON).

C. TRANSACTION LAYER

The transaction layer is devoted to let users share and trade their data. It is composed of a set of Ethereum smart contracts and of an Authorization Service, that act as intermediary for the user. Smart contracts are in charge of granting access to certain data. The access to a specific datum or a feature channel, indeed, is purchasable using dedicated smart contract methods. In Ethereum, such methods are payable functions that enact monetary transactions. Due to the presence of smart contracts, no direct interactions are needed among the data owner and users interested in his data.

1) FEATURES CONTRACTS

Each feature channel in IOTA is associated to a specific feature contract in Ethereum. Feature contracts allow for different types of access, each one at an associated crypto-monetary cost. In essence, the feature contract maintains an

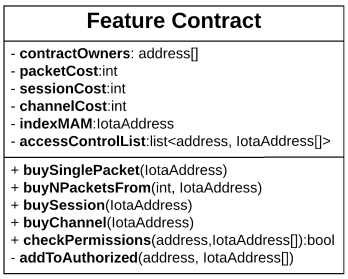

FIGURE 5. Feature contract represented as a class. Constructor and get/set methods are omitted.

Access Control List (ACL) that represents the rights to access the user's data. This list associates an Ethereum address to a bundle of data. This bundle is composed of the addresses of IOTA transactions containing a single datum (or a reference to an IPFS object), and by references to feature channels.

All the feature smart contracts have the same behaviour. They are created and instantiated in the Ethereum blockchain during the user ''registration'', thanks to the Factory design pattern. In particular, a Factory contract is used for creating ''child'' contracts and for storing their addresses, so that they can be extracted whenever necessary.

Figure 5 shows the class diagram of a Feature contract with attributes and methods. Attributes include the ACL and their goal is to provide a sort of ''menu'', with costs for accessing data. Whilst, methods are used to convey the access transaction or to check the access permissions.

2) AUTHORIZATION SERVICE

The authorization service is in charge of enforcing the access rights that are specified in the Feature contract ACL and to release the access keys to access data. A data consumer can send a request to the authorization service, in order to obtain the secret keys needed to decrypt messages in MAM Channels and IPFS Objects. Upon user request, the authorization service can check if he is eligible, through interaction with the smart contract. If this is the case, the authorization service provides the user with the related access keys. In particular, for each MAM message and its related IPFS object (if available), there is a key, which is used to encrypt the produced data.

The presence of an authentication service is necessary for two reasons: i) it is unfeasible to assume that every user is always on to receive access requests; ii) it is not possible that smart contracts can be autonomous in releasing decryption keys or decrypting messages, due to the fact that their computation is public.[1]

There are several methods to design this service. The main schemes are:

- **Client/Server** - This is the most feasible solution, i.e. one server provides the authorization service and holds the entire set of secret keys to access MAM messages and IPFS objects. The data consumer contacts the server directly to retrieve the keys he is eligible to get. This design implies that users trust the server, since this entity has the complete access to user data.
- **Proxy Re-Encryption (PRE)** - This solution is inspired to [68], where a single-use, unidirectional and not transparent PRE scheme for secure distributed storage is described. According to this approach, data blocks are encrypted using a content key k_c and then stored in a block store together with a lockbox, i.e. the content key encrypted using a master public key pk_m. In our scenario, the block is the MAM message, while the Tangle is the block store. When a data consumer wants to decrypt a block, he asks to the server of an authorization service to access the block through his public key. If the server has the necessary re-encryption key, it re-encrypts the lockbox and returns the new ciphertext. The re-encryption key is generated by the data owner using the consumer public key. The consumer, then, can decrypt the re-encrypted lockbox and obtain the key k_c. Since the server is required to be semi-trusted, a variant based on a consensus network has been proposed by the authors in [69], [70] for a decentralized scenario. This PRE solution is expected to have good performance (for example, with respect to the secret sharing approach explained below). However, it is achievable only when the data owner device is able to reply to re-encryption keys requests. As explained before, we cannot assume that the users' devices are always on. Therefore, this approach is not always suitable in our scenario.
- **Secret Sharing** - We resort to the Secret Store feature provided by the Parity Ethereum client [71], a popular Ethereum blockchain client. The distributed key management system is built using an overlay of Parity nodes. In particular, the Secret Store allows users to store fragments of the ECDSA (Elliptic Curve Digital Signature Algorithm) key along the overlay. A (t, n)-threshold scheme is employed. Thus, single nodes alone are unable to reconstruct an ECDSA key, because they only save a portion of this key. The retrieval of these fragments, that constitute the secret key, is controlled through a specific ''authorization'' smart contract interface [71]. The Feature contract implements this interface.

All these three techniques are supported in our software architecture. Due to their characteristics, we claim that ''secret sharing'' is the best choice. Such functionality is implemented within the *checkPermissions()* method of the Feature contract, shown in Figure 5. This method is called by the Parity Ethereum nodes during the authorization process [71].

D. SERVICE LAYER

This layer includes all the decentralized ITS services made available to users. They exploit all the functionalities and data produced in layers below. Smart services are based on the

[1] As a matter of fact, techniques have been proposed to execute private computation in blockchains, but this is out of the scope of this work [66], [67]).

trusted functionalities provided by smart contracts. Services use crowd-sensed data, whose reliability is founded on the use of certificates, i.e. proofs on data validity. In this work, we specifically focus on the authenticity of the geographical location of the produced data, which is a principal certificate to be provided in an ITS environment.

1) DATA AUTHENTICITY FOR PROOF OF LOCATION

One of the main concerns, related to the retrieval of crowd-sensed data, refers to data veracity and accuracy [72]. As a matter of fact, data is often uncertain, imprecise and difficult to trust. There is a number of related problems, ranging from issues related to accuracy, the presence of noise in the data sensing, up to voluntarily malicious user behavior, such as data falsification and so on. In this work, we will not deal with issues concerned with the accuracy of sensors [73]. Rather, we focus on a scheme that ensures that sensed data are securely added to a decentralized ledger, thus avoiding that an external entity might tamper such data. Indeed, once added into a DLT, data cannot be modified, due to the DLTs' data immutability features. Thus, the moment when the data can be altered is in between its generation and its insertion into the distributed ledger. To cope with this, in this context trust can be obtained through third party authority and proofs mechanisms. In particular, certificates are used as proofs of a certain user state property in space and time. Such certificates can be attached together with data in feature channels. Certificates are released in a system that can prove to be trusted, e.g. a Public Key Infrastructure (PKI), and which allows verifying some intrinsic properties of the produced data, e.g. geospatial coordinates.

Surely, in a mobility scenario, location covers an important feature, since it enables the implementation of useful context-aware services. An example of a proof-of-location certificate, that might be released in an ITS, is a bus ticket signed by the bus OBU, i.e. the bus certifies that the owner of the bus ticket was in that specific bus at a given time, and hence the user was in a certain geographical position at that time. Thus, the bus ticket may represent a proof-of-location certification authority, assuming that there is some trust on the bus company and on the data produced by its buses.

In general, Proof-of-Location (PoL) verifies the correctness of a user's claim to be in a certain position at a certain time [74]. We consider three ways to provide PoL certificates:

- **Use of PKI trust through Certificate Authorities (CA)**: any device operating on road (e.g. RSUs or public transport vehicles) that is trusted by a CA can be then trusted when releasing certificates. Since communication, through WiFi or Bluetooth technologies, requires spacial closeness, a user communicating through these technologies can issue a claim to this trusted device, that in turn can answer by releasing a signed certificate containing the device location. This certificate proves that the user is in the range of that trusted device, with an accuracy in the order of tens of metres. A PKI can be also built in a decentralized environment [75].
- **PoL trust through a trustless decentralized system**: FOAM [76] is an example of an open protocol for decentralized and geospatial data markets in which PoLs are created using trustless devices. FOAM is a permissionless and autonomous network of radios that offers secure location services through time synchronization, independent of centralized sources like GPS. Anyone can access to the network using his radio, e.g. LoRa, and he is incentivized by the protocol to cooperate with other participants in order to provide PoL to other users by using triangulation methods.
- **Zero Knowledge Proof of Location**: Zero Knowledge PoL (zk-PoL) allows a verifier to test whether a position committed by a prover is inside or outside the radius of a service area without revealing prover's exact location. With respect to the two previous approaches, zk-PoL introduces the concept of privacy, since it demonstrates that a user is within a certain area, without revealing the exact position of the user. Examples of proof-of-location protocols that enhance location privacy are shown in Platin [77] and in [27].

2) SMART SERVICES

With the term Smart Services, we refer to services built on top of the smart contracts presented above (feature contracts) and possibly novel ones, specifically thought and customized to support a specific ITS application. An example of smart service is a mobility tracking system, based on CAM, that can be used by insurance companies to identify vehicles responsible of road accidents, in order to simplify the resolution of conflicts [1]. Another use case could be the development of a public transport monitoring application, based on individuals' witnesses (similarly to the scenario we simulate in Section V).

An important aspect of smart services consists in payments, that must occur when data are traded among different parties. We use a standard ERC20 token to perform transactions within the ITS. This token is involved in all ITS related payments, from data access rights to micropayments. Having a unique token, used and shared among all ITS services, can be beneficial for a global use of all the services offered in the ITS, e.g. a user that receives a token for having shared some data, then uses it to buy a bus ticket.

We consider two kinds of smart services, based on their implementation: ''smart contract based services'' and ''on road services''.

a: SMART CONTRACT BASED SERVICES

Users, that are able to provide data or services, can perform transactions with other users through smart contracts, that are specifically designed for a given service. The service payment is accomplished directly on-chain, as soon as contract methods are called. Service providers can use data provided by users to acquire knowledge of a particular area. Such data

foster the development of geo-localized smart services, built using smart contracts as business logic. Thus, for example, a user A may be both a data provider and a service consumer, whereas a user B provides a service by consuming data (that can be generated by users like A). User B gains access rights to some data through feature contracts owned by user like A and then offers a service based on that data to A. Such service might be offered in exchange of a payment; the payment will be accomplished through a transaction, issued to a specific smart contract related to the service and owned by B.

b: ON ROAD (DIRECT COMMUNICATION) SERVICES

On road services are direct communication services operated by devices/vehicles in the ITS. These services exploit V2X connections, where "X" indicates a device that is capable of offering Smart Services and the connection happens through short range communications. These services are mainly based on micropayments. Examples are:

- Parking and bus tickets - micropayments are exchanged through a NFC communication channel with the ending device.
- Limited traffic area and motorway tolls payments - messages are exchanged through WiFi Direct passing through gates.
- Pay-per-drive - pay-per-use car rental services, made possible through communication between the AU and the vehicle's OBU, based on the traveled distance.

To foster the implementation of such services, we implemented a micropayment channel network, based on the μRaiden framework [22], [23]. μRaiden is an open source framework, used to implement ERC20-based free pay-per-use payment channels on top of the Raiden Network. The rationale behind this choice is that it is not feasible to open a novel payment channel, every time a user interacts with an on road service. To avoid this, Raiden Network allows creating an overlay network of payment channels. Thus, if, for instance, a user A has an opened channel with user B, and B has an open channel with C, then A can pay C through B, without having to open a novel payment channel between A and C.

IV. A SYSTEM USE CASE TOWARDS A DATA TRADING AND PRIVACY

Data in ITS are usually shared among many independent and self-interested participants, each of them giving a specific value to such data and then leading to the creation of a data marketplace. The data marketplace we envision [78] allows connecting data providers and consumers, ensuring data high quality, consistency and security. Data providers are recognized as owners in the marketplace and receive benefits (mostly economical) in exchange of the datasets and data streams they supply. Consumers pay for data they acquire and may provide new information back to the marketplace, in return. Smart contracts (e.g. feature contracts) can automate the negotiation among data providers and consumers, providing advantages for both parties.

Clearly enough, such service must guarantee privacy, when needed, in order to avoid the disclosure of sensitive information.

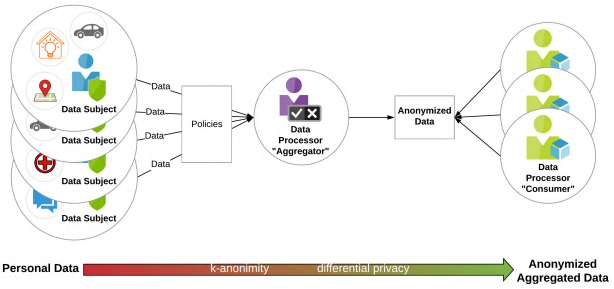

FIGURE 6. Sensing-as-a-Service in a data marketplace.

A. SMART SERVICE: ANONYMIZING DATA BY AGGREGATION

In the marketplace, data providers maintain their data, as described in the previous sections, and register data they want to sell, along with standardised descriptions of what they measure, i.e. ontologies. (In this work, we will not go into the details of aspects related to data description, giving more focus on the system architecture.) Such data, coming from different sources, can be aggregated to provide anonymized datasets, that might be more valuable than single data points.

Figure 6 shows an example of how such Sensing-as-a-Service can be implemented [79]–[81]. An entity takes the role of data processor, standing between providers and consumers: such "aggregator" gathers data from individual data providers and produces anonymized aggregated data, ready to be acquired by consumers. The main steps of this process are as follows:

- Data providers release data measured in the ITS to the aggregator, by granting the access on the related Feature contracts. An Aggregation contract regulates policies, rights and obligations of both aggregators and providers. Hence, by invoking the contract's functions, all participants agree to these policies.
- The aggregator selects k providers, among those available, using the quality of the data as the discriminant, i.e. Proof of Sensing [78]. These providers become members of a k-DAO (Decentralized Autonomous Organization [82]) created to limit possible aggregator's malicious behaviors. The aggregator, indeed, stakes a safety deposit, and the k-DAO at any moment can decide to vote (through the smart contract) to redeem this deposit, if the aggregator misbehaves.
- Finally, data is aggregated in a dataset that presents properties of k-anonymity [83] and differential privacy [84]. This dataset can then be acquired by data consumers in a process where every participant is rightfully rewarded (directly though a contract method).

This service is specifically thought to protect the privacy of data providers and to produce anonymized data. Not only, it also provides some advantages in a data marketplace. For instance, through this approach the consumer can gather large quantities of the same kind of data, rather than searching single providers, one by one. Similarly, data aggregation is an advantage also for the data sellers, since they have more chances to sell their data in an aggregate form.

V. PERFORMANCE EVALUATION

The aim of this section is to assess the performance and scalability of the proposed system. This system is expected to attend a large number of clients that publish data to be used for smart services or to constitute a marketplace. In this sense, surely, the most critical part of the system architecture is on the use of DLTs, i.e. Data and Validation layers of Figure 3. In particular, we are interested in evaluating the goodness of adopting IOTA as the immutable registry for ITS, as well as the DFS solutions. Thus, we focused on the transmission of sensed data to the system presented above, measuring latencies needed to issue, insert and validate messages/transactions (TXs), and also the level of reliability of the network nodes.

A. SETUP: TRACE-DRIVEN VEHICLES SIMULATION

Our experimental scenario was based on a hypothetical real ITS application. In particular, we conducted a trace-driven experimental evaluation. Traces were generated using the RioBuses dataset, a real dataset of mobility traces of buses in Rio de Janeiro (Brasil) [85]. Based on these traces, we simulated a number of buses that, during their path, periodically generate sensed data. These data may represent temperatures, air pollution values, etc. Here, we are mainly interested in the behaviour of the system, hence we focus only in one type of datum, i.e. the bus geolocation (latitude and longitude). We assume that the time spent to fetch such data is negligible, with respect to the time to publish it. Figure 7 shows the paths of 10 buses, as an example, that were considered during our tests.

These messages were utilized to generate real requests transmitted to both the DLT and the chosen DFS. Each message was sent to a given node of the networks.

- *DLT tests*: we mainly focused on IOTA node selection (next subsection) and on its scalability. We varied the number of buses in the range: 60, 120, 240. For each bus, we utilized one hour of trace data. Based on the paths, each bus was set to generate approximately 45 message/hour. Thus, we made one hour long tests, where each bus generated, on average, a message to be issued to the DLT every 80 sec, which is a reasonable time interval to sense data in an urban scenario. For each test configuration, we replicated the experiment 12 times.
- *DFS tests*: we compared two DFSes solutions: IPFS and Sia [20]. The idea is to study the solutions to store data in a DFS, comparing the latencies to request IPFS nodes and Sia nodes. In the case of IPFS, we assessed two different scenarios: i) the case with a dedicated IPFS node, devoted to handle only requests coming from our application (referred as "IPFS Private"), ii) the case with a public IPFS node, that can be contacted also by other applications (referred as "IPFS Public"). In the case of Sia, we exploited the Skynet platform services [86] to easily access the provided permanent storage. A limitation of these assessments was the usage of a single DFS node. Thus, in order to keep a comparable workload between the DFS and the DLT tests, the number of buses was set equal to 10, i.e. 10 bus for each node on average (Section V-B3).

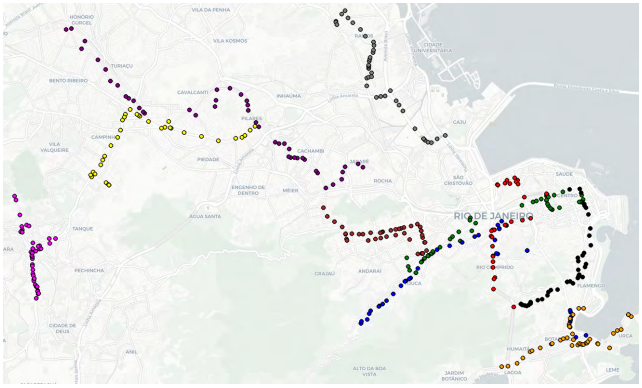

FIGURE 7. The 1 hour long path of 10 buses in Rio de Janeiro (Brazil).

For each request, we recorded the outcome, i.e. successful or unsuccessful, due to some nodes internal error, as well as the latency between the transmission of the request (a TX in the case of IOTA) and the confirmation of the data being published in the network.

1) IOTA SETUP

Each bus was emulated by a single process (issuing messages based on the data trace). Thus, the first task was to find, for each bus, a full node of the IOTA DLT to interact with. In our tests, we were enabled to rely only on services that maintain a public list of active nodes [87]. This is because IOTA network full nodes do not usually allow to list their neighbors in the P2P overlay, through API. Hence, it is not possible to perform an in-depth graph search on the overlay and to retrieve an up-to-date list of active nodes to interact with. With this in view, the scheme we designed, to select the IOTA nodes to contact, is as follows. Given the list of public nodes, a filter is applied to keep only nodes that are fully synchronized, i.e. the node has solidified all the milestones up to the latest one released by the coordinator, and that allows remote PoW. During our tests, these IOTA nodes were ~ 60. Then, we designed three heuristics for the selection of a full node from the public pool:

1) **Fixed Random**: Each bus is assigned to a random IOTA full node from the pool, during the setup phase; then, every TX generated by that bus is handled by this node, for the whole duration of the test.

2) **Dynamic Random**: A random node from the pool is selected every time a message has to be published by a bus.
3) **Adaptive RTT**: For each bus, its associated node actively changes every time a message has to be published, while the previous one is still pending. Based on results of past interactions, the known IOTA nodes are ranked through the experienced Round Trip Time (RTT) [88]. Then, a new node is chosen by selecting the best known node or, if every known node is in the process of publishing a message, a new node is picked randomly from the pool.

We used a MAM channel associated to each single bus. The procedure followed is shown in the algorithm in section III-B2 (in particular, lines 7-11). It is worth noticing that the use of a MAM channel significantly increases the latencies to add a TX in the ledger, with respect to single messages. In fact, each message to be published in the MAM channel requires three TXs to be issued, i.e. one containing the data and two other messages for the signature. The creation of such bundle of TXs requires, on average, 1475 msec using a smartphone considered as the AU (Qualcomm Snapdragon 625 MSM8953 CPU), while 224 msec using a server (Intel Core i7-6700HQ CPU).[2] However, after the creation of a message, during the communication with the provider the two hardware configurations were not different in performance; therefore, all the reported measurements are based on the server setup.

The use of a MAM channel for each bus has the advantage to permit a fast and easy retrieval of each bus's data stream. For each TX, we measured the time required to perform the tip selection, as well as the PoW. The tip selection depth parameter, i.e. the number of milestones to go back to start the random walk to select tips, was set to 3, whilst the minimum weight magnitude, i.e. the number of trailing zeros of a TX hash, was 14 (minimum standard value for the IOTA mainnet).

2) IPFS & SIA SETUP

As well as in the IOTA tests, the same data-driven simulation was employed to simulate the buses' behavior. However, in this case, each process (modelling a bus) interacts with only one node, which can be of 3 types:

- **IPFS Private Node**: We setup an IPFS node on a dedicated device, connected to other nodes in the main network. Thus, the host simulating the buses was the only one sending requests to it. The files are stored locally (and on its IPFS neighbors) and as long as someone is incentivized to "pin" it, i.e. to keep it, it remains available to anyone.
- **IPFS Public Node**: The previous case requires that every user maintains a proprietary IPFS node. Actually, it is more reasonable for users to rely on a service provider. With this in view, we tested the Infura service [64], that offers a free access to IPFS.
- **Sia Node**: Tests are conducted making requests to a node in the Skynet, a content delivery platform built on top of Sia. A Skynet node (webportal) is a special Sia node that has already formed contracts with every available host, paying for the files uploaded to them, and thus proposing a service with its own policies on how many and what types of files you can upload [86]. The estimated price for storing files in Sia is reasonably low, i.e. around $2/TB/month (when the network is fully optimized) [89].

Tests consist in measuring the latency to successfully complete a message upload. To foster the reproducibility of our experiments, the entire dataset (comprehending IOTA, IPFS and Sia) and the implemented scripts are made available in a public github repository [90] with a Free Software license.

[2]We think that performances on smartphones can be further enhanced, since the used libraries for MAM channels were not native.

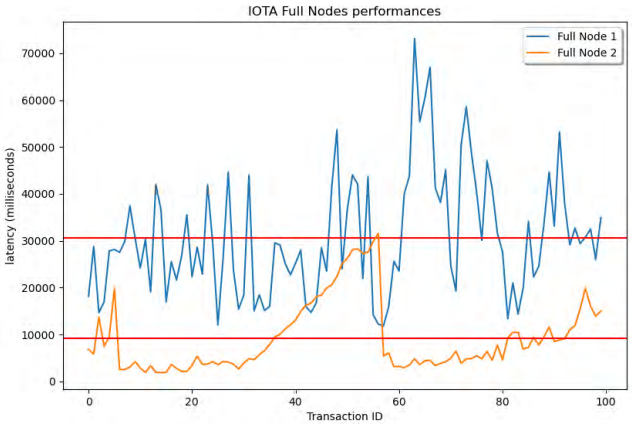

FIGURE 8. Latency comparison between two IOTA full nodes attaching simultaneously a sequence of 100 TXs.

B. RESULTS: IOTA

1) NODES PERFORMANCE DIFFERENCE

A first important result to mention is the wide difference on the performance of different IOTA full nodes. As an example, Figure 8 shows latencies we measured using two exemplars IOTA full nodes. As the figure shows, there is a significant difference on the times required to perform the same task. This is probably caused by the different nodes' hardware capabilities, as well as the possibly different workload they were subjected to, during the tests. In fact, IOTA full nodes are not homogeneous, nor there is some kind of load balancing mechanism that coordinates the requests received from users. Not only, it is important to point out that a node with a higher network degree may receive more TXs from others; therefore, it is faster to provide tips to clients.

In particular, this test consists in attaching 100 TXs to the Tangle, but *Full Node 1* is relatively "unknown", while *Full Node 2* is one of the most used nodes that operate in IOTA. This means that *Full Node 1* has more resources to provide to clients but less tips, since it receives less TXs, while *Full Node 2* is faster in providing tips but it has a limited amount

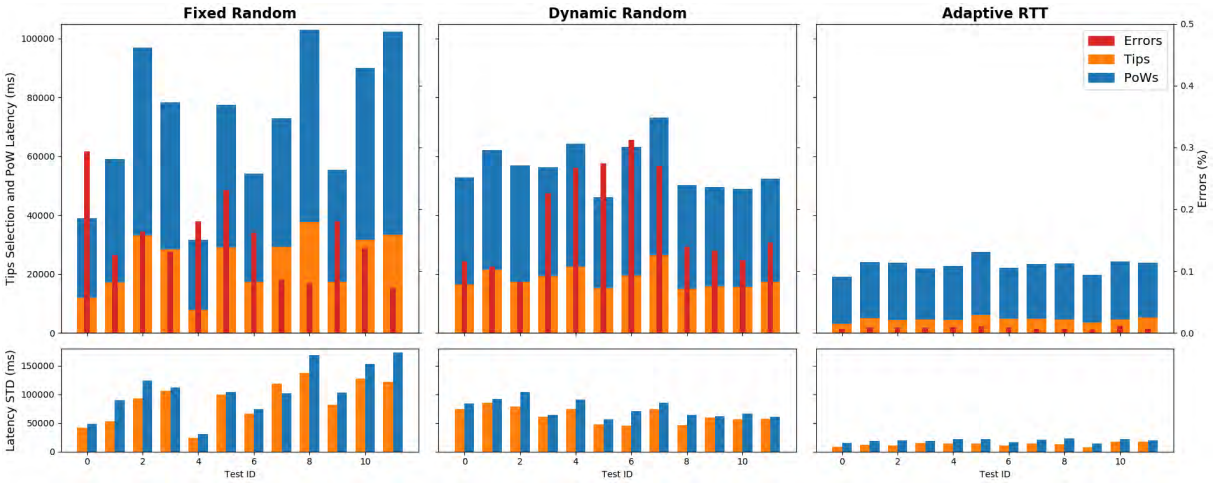

FIGURE 9. 60 bus tests: average latencies, standard deviation and errors for the three different schemes (lower is better).

of resources to devote to each client. The result is that *Full Node 2* allows to attach a TX in 9 seconds on average, while *Full Node 1* needs more that 30 seconds.

These preliminary results demonstrate that the choice of the used full nodes is relevant. In the rest of the section, we show results related to the different heuristics, employed to select the IOTA full nodes.

2) NODE SELECTION

Figure 9 shows results obtained for different test repetitions, when the number of emulated buses was set to 60. In particular, we show the results for each scheme we employed for the selection of the nodes. In the upper part, the bar charts report the average latencies measured during a single test. The orange (lighter) part of the bar chart shows the average latency to perform the tip selection, while the blue (darker) part shows the average latency associated to the PoW. The red (central and smaller) bars refer to the percentage of errors (the related y-axis is shown on the right of the figure), i.e. amount of TXs that failed to be added to the Tangle, due to full nodes' errors. On the lower part of the figure, we show the average standard deviations related the specific tests, both for the tip selection and PoW. From the figure, it is possible to appreciate how in general a random selection of the full node to issue a TX does not lead to good results. The amount of errors is quite high, as well as the measured latencies. Thus, these tests seem to conclude that, at the time of writing, the IOTA DLT is not fully structured to support smart services for transportation systems. On the other hand, the good news is that if we carefully select the full node to issue a TX, the performances definitely improve. In fact, our third scheme "Adaptive RTT" has a low amount of errors, on average around 0.8%. Measured latencies are lower than other approaches because well performing nodes are chosen more often. Still, the average latency amounts to 23 seconds, which is far from a real-time update of the DLT. The level of

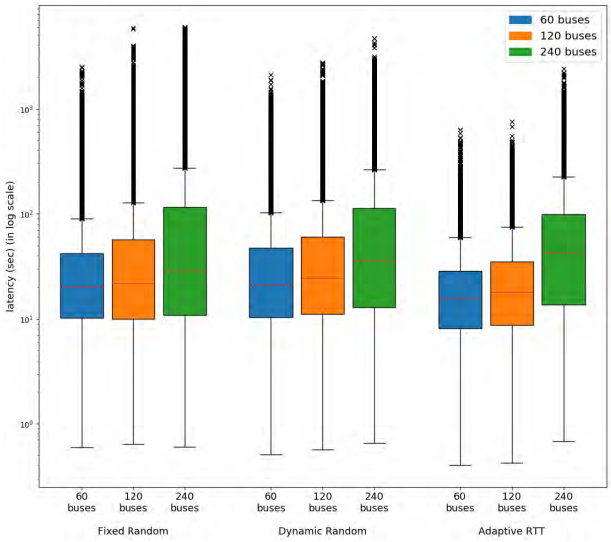

FIGURE 10. Latency (in log scale): boxplots for tests with 60, 120, 240 buses.

acceptability of latency values truly depends on the application scenario.

These first results suggest that some scalability tests might give further insights on the viability of the use of IOTA as the DLT to support smart transportation system. For this reason, we made some tests with an increasing number of buses.

3) DLT SCALABILITY

Figure 10 shows average results obtained using our three considered schemes, when varying the number of buses. Results are reported as box plots. Thus, each box plot corresponds to the average results for a scheme in a given scenario. This allows to assess the scalability of each scheme, by looking at the results for an increasing amount of buses. At the same time, it is possible to compare the three schemes by looking at their performance for each scenario.

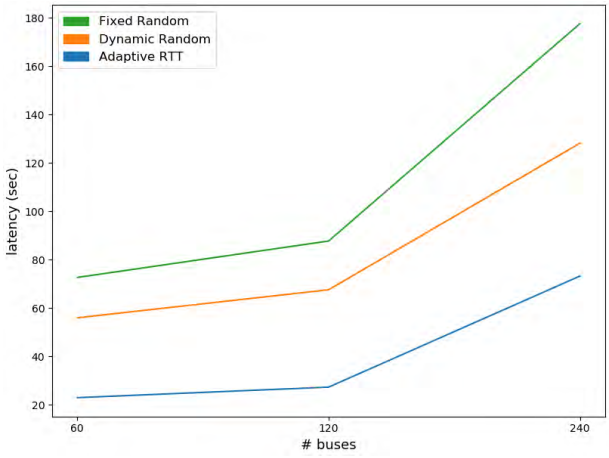

FIGURE 11. Average latencies increasing the number of buses.

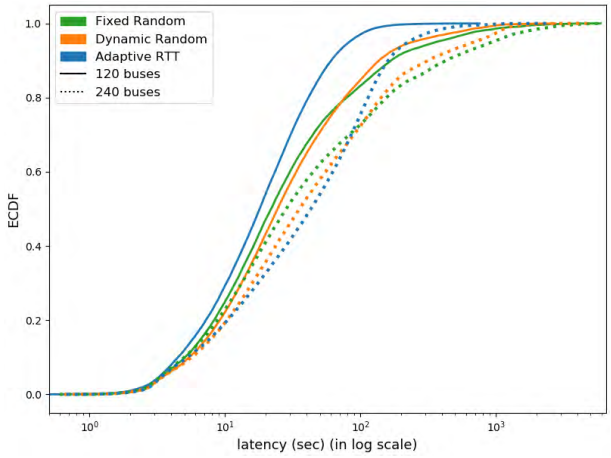

FIGURE 12. Empirical cumulative distribution function for tests with 120 and 240 buses.

The rectangle identifies the Inter-Quartile Range (IQR), i.e. values from the 25th to the 75th percentile, representing the middle 50% of values. Hence, the lower part of the box (let denote it Q1) is the first quartile (25th percentile), the highest (denote it Q3) is the third quartile (75th percentile). The red line inside the box is the median value. The lower and upper values identified by the vertical line are the whiskers. In box plots, the whiskers are defined as 1.5 times the IQR. Thus, the lower whisker is Q1 - 1.5*IQR, while the upper whisker is Q3 + 1.5*IQR; they represent a common way to describe the dispersion of the data. Finally, the "×" symbols outside the whiskers are the outliers. To better show the obtained results, the y-axis is reported in a log scale.

Results show that in all cases, average latencies increase significantly with the number of buses. It is worth noticing that, being the y-axis in log scale, the difference on the performance is relevant. It is also confirmed that "Adaptive RTT" provides better results since average latencies are definitely lower than other schemes. In particular, the first two schemes have outliers well over 10^3 sec. This suggests that the number of full nodes devoted to the TX management should increase proportionally to the number of buses.

A set of issued TXs is handled by each node sequentially, i.e. one after the other. Thus, a delay occurred in a TX have repercussions on subsequent ones, in terms of latency to add TXs in the DLT. For instance, when 240 buses are active and send requests, we have a message generation rate of about $\sim$ 3 msg/sec, to be issued to the IOTA DLT. If we assume that the workload is evenly distributed among 60 nodes, then, each node handles TXs from about 4 buses, thus receiving, on average, a new TX request every $\sim$ 20 sec. Bearing in mind that, at best, it takes 23 sec for a full node to process a TX, then we see that an initial overhead of a few seconds leads to a huge increase at the end of the test. It is worth noticing, however, that in "Adaptive RTT", the same $\sim$ 15 full nodes were utilized (as they performed better than the others), with an average workload of 16 buses per node, which means a new request every 5 sec. For this reason, the latency increased to 73.26 sec on average. This means that further improvements are needed to solve scalability issues.

To better emphasize the outcomes, Table 1 reports some summarized statistics (shown in the box plots) and the error rates. Actually, the studied approaches show a great difference in the amount of errors. While the average error for "Adaptive RTT" is $\sim$ 1%, for the other two schemes we have errors well above 15%. These error rates are clearly unacceptable, meaning that these approaches are unusable in real ITS scenarios.

Finally, Figure 12 shows the empirical cumulative distribution function obtained for the compared schemes in the 120 and 240 bus scenarios. In this case, for the sake of a better visualization, the x-axis is in log-scale. These charts further confirm the better performance obtained by the "Adaptive RTT" scheme.

4) ALTERNATIVE APPROACH: LOCAL PoW

In this section, we evaluate an alternative approach, which consists in adopting an edge computing system model. In this case, the PoW is executed by the gateway of the edge computing system, e.g. an RSU. (The tip selection must be always accomplished at a full node, that maintains a complete copy of the Tangle.) The rationale would be to relieve the IOTA node from the computational burden of the PoW. However, this would force to equip the gateway with sufficient computational capabilities to perform the PoW for all the TXs generated by the buses it handles.

We made some tests using a host, as the gateway, equipped with a NVIDIA GTX 950 GPU (the PoW algorithm is executed on GPUs). The host was in charge of handling messages generated from a different amounts of buses, i.e. 1, 5, 10 buses. Table 2 shows the average times needed to perform the PoW at the gateway for 1, 5 or 10 buses (these latencies are related to the actual time required to perform the PoW), and compared to results obtained with a public IOTA full node for 10 buses (in this case, the latency includes the RTT for the

TABLE 1. Results on IOTA, with 60, 120, 240 buses.

# buses	Heuristic	Avg Latency	Conf. Int. (95%)	Errors
60	Fixed Random	72.68 sec	[70.43, 74.94] sec	15.37%
	Dynamic Random	56.0 sec	[54.51, 57.5] sec	18.26%
	Adaptive RTT	22.99 sec	[22.69, 23.29] sec	0.81%
120	Fixed Random	87.75 sec	[85.38, 90.12] sec	29.49%
	Dynamic Random	67.6 sec	[66.29, 68.9] sec	18.99%
	Adaptive RTT	27.35 sec	[27.11, 27.58] sec	1.1%
240	Fixed Random	177.62 sec	[174.25, 181.0] sec	42.81%
	Dynamic Random	128.2 sec	[126.28, 130.12] sec	44.85%
	Adaptive RTT	73.26 sec	[72.68, 73.85] sec	7.55%

TABLE 2. Comparison between PoW performed at the gateway (local) and by delegating a IOTA public node.

# buses	Avg Latency	Conf. Int. (95%)
(Local) 1	12.61 sec	[11.33, 13.88] sec
(Local) 5	22.76 sec	[21.29, 24.23] sec
(Local) 10	71.07 sec	[66.12, 76.02] sec
(Delegated) 10	18.01 sec	[16.24, 19.76] sec

communication with the full node). In this case, results show that a public node is definitely faster (when it is well chosen). This means that adequate hardware nodes are needed, and this results in a trade-off that must be considered in a comprehensive cost/benefit evaluation. The more computationally efficient these nodes are, the more they will be able to handle PoW requests coming from vehicles. Similarly, the higher the density of these gateways in a given geographical area, the lower their workload. Thus, in the design of an edge computing platform, the optimal placement of these gateways should be made carefully, based on their hardware characteristics.

C. RESULTS: IPFS & SIA

In this section, we show results for the tests with the DFS. The box plots in Figure 13 show the average results obtained when making requests to the IPFS Private node, the IPFS Public Infura node and the Sia node. These results relate to messages generated from 10 buses.

Results show that the difference on the performance is relevant between the private case and the public ones. This was expected, since the private node processes only requests coming from the buses, while the other two nodes receive multiple requests from other unknown clients simultaneously. IPFS Private has a mean latency of about 1 sec, with a very limited standard deviation. We experienced some outliers in the latencies, that nevertheless where all below 9 sec. Conversely, IPFS Public and Sia do have a wide latency dispersion. Even if the Skynet-Sia node has volatile performances, on average it performs better than the Infura-IPFS one. In both cases, the latency outliers reach values above 18 sec. In particular, in the chart we limited the *y*-axis to 18 sec. However, in some cases we measured latencies above 41 sec for IPFS Public, and 25 sec for Sia.

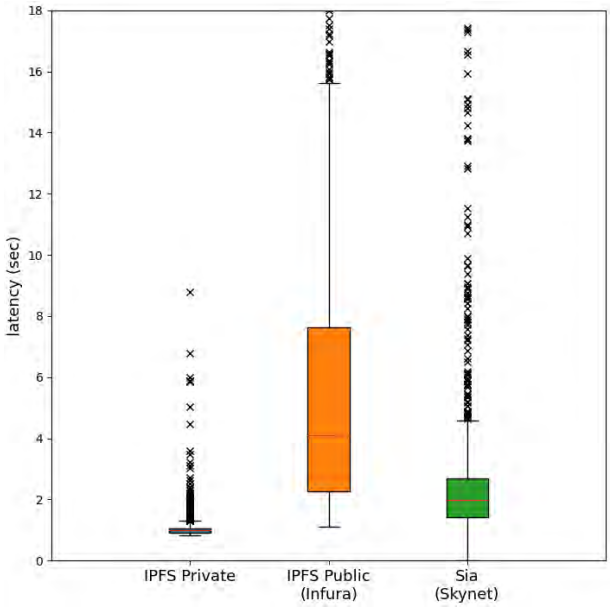

FIGURE 13. Latency: boxplots for tests in IPFS and Sia. (Limited at 18 sec for clarity).

TABLE 3. Results on DFS.

Case	Avg Latency	Conf. Int. (95%)	Errors
IPFS Private	1.07 sec	[1.05, 1.09]	0%
IPFS Public	5.48 sec	[5.27, 5.69]	1%
Sia	2.41 sec	[2.31, 2.51]	0%

As already mentioned, we had a limited access to the DFS nodes. This limited our tests to the use of a single node that handles the buses requests. This in turn limited the amount of buses requests and, in order to keep the requests associated to each node comparable to those of DLTs tests, we kept the amount of buses equal to 10. Results show that, as expected, DFSes are definitely faster than DLTs, since there are no issues concerned with tip selection and PoW. But, of course, their role in the ITS ecosystem is different. DFSes only maintain data, and they can be utilized to provide data availability and reliability guarantees, while DLTs provide immutability guarantees. Thus, these two technologies must be provided in conjunction (as we do in our system architecture).

VI. DISCUSSION

As already mentioned in the previous section, results on the employed DFSes show good performance. The latencies measured to add data into IPFS and Sia are practically acceptable in ITS scenarios. Clearly enough, an adequate ITS infrastructure must be set, in order to build a scalable architecture that is able to properly handle a possibly high data generation rate from multiple moving vehicles. Put in other words, we think that the issue is concerned more on the system deployment, rather than on the DFS protocol. For instance, an edge computing architecture can be used to geographically place DFS node gateways, which receive data from vehicles and insert them into the DFS.

On the other hand, our results show that the design of effective, responsive and reliable DLTs is a crucial aspect. For instance, if IOTA is chosen as the DLT for storing data, then it is important to properly select the IOTA nodes to interact with, in order to get error rates that are acceptable. However, measured latencies resulted higher than 20 sec, which is quite high if we think at real-time applications, but reasonable for less time demanding services. Regardless of the proper selection of the faster full node, an important part of the time required for the validation of transactions is due to the execution of the PoW. This time can be lowered by asking to a computationally efficient node to perform this task. However, it cannot be eliminated, in IOTA.

With this in view, it is important to mention that we conducted preliminary tests with other possible DLTs, that implement novel techniques to improve responsiveness and scalability. Among the others, a solution worth of mention is Radix, a novel DLT that implements sharding techniques [91]. In few words, sharding consists in breaking the ledger into smaller, more manageable chunks, and distributing those chunks across multiple nodes, in order to spread the load and maintain a high throughput. At the time of writing, the Radix technology is still in its infancy and a proper main net does not exist, yet. Nevertheless, we exploited the alphanet test network to issue transactions on the ledger. Thus, obtained results cannot be considered accurate and it is too early to give an overall judgment on this DLT. However, we obtained very low latencies (below 1 sec), with a non negligible (but low) error rate. We stress the fact that these results cannot be compared with those obtained for IOTA. In fact, in IOTA we exploited the main net, while in Radix we had to employ a preliminary testnet, with few nodes involved to the ledger management ($\sim$ 6 nodes) and basically no additional workload, apart from our tests. As a matter of fact, comparable results can be obtained if tests are executed on the IOTA test net, where the PoW is faster (we obtained average latencies around $\sim$ 2 sec).

To conclude this discussion, we mention that, even if there are some novel interesting proposals to improve the scalability of DLTs, such as sharding or the Ethereum plasma [92], a main problem refers to the high fees that are often associated to every transaction. In fact, IOTA is designed to be feeless. Conversely, Radix and Ethereum are based on fees. These costs may be acceptable only when the transaction fees are negligible with respect to the value of the data.

VII. CONCLUSION

In this paper, we presented a system architecture that is able to manage and exploit crowd-sensed data in the novel generation of ITS, to foster the development of novel smart and intelligent transportation services. Our architecture exploits DLTs and DFSes, to conveniently store and secure data. Moreover, certificates, such as Proof of Location, allow to authenticate such data. Ethereum smart contracts and a distributed key management system are employed to control data access and authorization.

Experiences with the implemented software architecture allowed us to conclude that the use of DLTs, together with DFS and sophisticated cryptographic schemes, permits to viably control data access and to offer interesting services, while maintaining data authenticity and availability. We have shown the case of a data marketplace, as an application example.

We claim that an important and critical outcome of this work is concerned with the experimental assessment we performed, and the related results of the current technologies available at the moment. Latencies measured to store data into the considered DFSes, i.e. IPFS and Sia, can be considered acceptable for general ITS scenarios. As concerns the employed DLT, i.e. IOTA, we conclude that at the moment the obtained results are not viable for real-time applications, but acceptable for less demanding services.

REFERENCES

[1] C. F. Chiasserini, P. Giaccone, G. Malnati, M. Macagno, and G. Sviridov, "Blockchain-based mobility verification of connected cars," in *Proc. IEEE 17th Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2020, pp. 1–6.
[2] C. E. Palazzi, M. Roccetti, and S. Ferretti, "An intervehicular communication architecture for safety and entertainment," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 1, pp. 90–99, Mar. 2010.
[3] (Apr. 2014). *Etsi en 302 637-2 v1.4.1*. [Online]. Available: https://www.etsi.org
[4] (2020). *Mobi: Mobility Open Blockchain Initiative*. [Online]. Available: https://dlt.mobi/
[5] (2018). *Dovu Whitepaper*. [Online]. Available: https://www.dovu.io/whitepaper.pdf
[6] S. Nakamoto. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf
[7] V. Buterin. (2013).*Ethereum White Paper*. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper
[8] S. Ferretti and G. D'Angelo, "On the ethereum blockchain structure: A complex networks theory perspective," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 12, Jun. 2020, Art. no. e5493.
[9] J. Benet, "IPFS–content addressed, versioned, P2P file system," 2014, *arXiv:1407.3561*. [Online]. Available: http://arxiv.org/abs/1407.3561
[10] J. Brogan, I. Baskaran, and N. Ramachandran, "Authenticating health activity data using distributed ledger technologies," *Comput. Struct. Biotechnol. J.*, vol. 16, pp. 257–266, 2018.
[11] M. Zichichi, M. Contu, S. Ferretti, and G. D'Angelo, "LikeStarter: A smart-contract based social DAO for crowdfunding," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2019, pp. 313–318.
[12] G. D'Angelo, S. Ferretti, and M. Marzolla, "A blockchain-based flight data recorder for cloud accountability," in *Proc. 1st Workshop Cryptocurrencies Blockchains Distrib. Syst.*, 2018, pp. 93–98.

[13] R. Di Pietro, X. Salleras, M. Signorini, and E. Waisbard, "A blockchain-based trust system for the Internet of Things," in *Proc. 23nd ACM Symp. Access Control Models Technol.*, Jun. 2018, pp. 73–78.

[14] A. Elsts, E. Mitskas, and G. Oikonomou, "Distributed ledger technology and the Internet of Things: A feasibility study," in *Proc. 1st Workshop Blockchain-Enabled Netw. Sensor Syst.*, 2018, pp. 7–12.

[15] S. K. Pinjala and K. M. Sivalingam, "Dcaci: A decentralized lightweight capability based access control framework using iota for Internet of Things," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Apr. 2019, pp. 13–18.

[16] B. Leiding, P. Memarmoshrefi, and D. Hogrefe, "Self-managed and blockchain-based vehicular ad-hoc networks," in *Proc. ACM Int. Joint Conf. Pervas. Ubiquitous Comput., Adjunct*, Sep. 2016, pp. 137–140.

[17] M. Zichichi, S. Ferretti, and G. D'Angelo, "A distributed ledger based infrastructure for smart transportation system and social good," in *Proc. IEEE 17th Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2020, pp. 1–6.

[18] R. W. van der Heijden, F. Engelmann, D. Mödinger, F. Schönig, and F. Kargl, "Blackchain: Scalability for resource-constrained accountable vehicle-to-x communication," in *Proc. 1st Workshop Scalable Resilient Infrastruct. Distrib. Ledgers*, 2017, pp. 1–5.

[19] *Regulation (eu) 2016/679—Directive 95/46*, Council of European Union, Brussels, Belgium, 2018, pp. 1–88.

[20] D. Vorick and L. Champine. (2014). *SIA: Simple Decentralized Storage*. [Online]. Available: https://blockchainlab.com/pdf/whitepaper3.pdf

[21] S. Popov. (2016). *The Tangle*. [Online]. Available: https://iota.org/IOTA_Whitepaper.pdf

[22] (2019). *Movo Prototype*. [Online]. Available: https://github.com/miker83z/movoApp

[23] (2019). *Micropayments*. [Online]. Available: https://github.com/miker83z/Mycropayments

[24] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular ad hoc network," *J. Netw. Comput. Appl.*, vol. 37, pp. 380–392, Jan. 2014.

[25] E. Androulaki, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, Apr. 2018, pp. 1–15.

[26] L. Li, J. Liu, L. Cheng, S. Qiu, W. Wang, X. Zhang, and Z. Zhang, "CreditCoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2204–2220, Jul. 2018.

[27] A. R. Shahid, N. Pissinou, L. Njilla, S. Alemany, A. Imteaj, K. Makki, and E. Aguilar, "Quantifying location privacy in permissioned blockchain-based Internet of Things (IoT)," in *Proc. 16th EAI Int. Conf. Mobile Ubiquitous Syst., Comput., Netw. Services*, New York, NY, USA, Nov. 2019, p. 116.

[28] M. Bez, G. Fornari, and T. Vardanega, "The scalability challenge of ethereum: An initial quantitative analysis," in *Proc. IEEE Int. Conf. Service-Oriented Syst. Eng. (SOSE)*, Apr. 2019, pp. 167–176.

[29] F. Vogelsteller and V. Buterin, "Erc-20 token standard," in *Ethereum Foundation*. Cham, Switzerland: Zug, 2015.

[30] J. Poon and T. Dryja. (2016). *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*. [Online]. Available: https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-network-paper.pdf

[31] *Raiden Network*. (2020). [Online]. Available: https://raiden.network,

[32] D. Pan. (2020). *IOTA Foundation Suspends Network, Probes Fund Theft in Trinity Wallet*. [Online]. Available: https://www.coindesk.com/iota-foundation-suspends-network-probes-fund-theft-in-trinity-wallet

[33] IOTA Foundation. (2019). *The Coordicide*. [Online]. Available: https://files.iota.org/papers/Coordicide_WP.pdf

[34] J. Benet and N. Greco, *Filecoin: A Decentralized Storage Network*. London, U.K.: Protoc. Labs, 2018.

[35] S. Higgins. (2017). *257 Million: Filecoin Breaks All-Time Record for ICO Funding*. [Online]. Available: https://www.coindesk.com/257-million-filecoin-breaks-time-record-ico-funding

[36] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz, "Permacoin: Repurposing bitcoin work for data preservation," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 475–490.

[37] S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin. (2014). *Storj a Peer-to-Peer Cloud Storage Network*. [Online]. Available: https://storj.io/storj.pdf

[38] M. Naz, F. A. Al-zahrani, R. Khalid, N. Javaid, A. M. Qamar, M. K. Afzal, and M. Shafiq, "A secure data sharing platform using blockchain and interplanetary file system," *Sustainability*, vol. 11, no. 24, p. 7054, Dec. 2019.

[39] D. Hawig, C. Zhou, S. Fuhrhop, A. S. Fialho, and N. Ramachandran, "Designing a distributed ledger technology system for interoperable and general data protection regulation–compliant health data exchange: A use case in blood glucose data," *J. Med. Internet Res.*, vol. 21, no. 6, Jun. 2019, Art. no. e13665.

[40] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems," *IEEE Access*, vol. 6, pp. 38437–38450, 2018.

[41] U. Feige, A. Fiat, and A. Shamir, "Zero-knowledge proofs of identity," *J. Cryptol.*, vol. 1, no. 2, pp. 77–94, Jun. 1988.

[42] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1998, pp. 127–144.

[43] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[44] X. Wang, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu, and K. Zheng, "Survey on blockchain for Internet of Things," *Comput. Commun.*, vol. 13, pp. 10–29, Feb. 2019.

[45] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with IoT. Challenges and opportunities," *Future Gener. Comput. Syst.*, vol. 88, pp. 173–190, Nov. 2018.

[46] K. Biswas and V. Muthukkumarasamy, "Securing smart cities using blockchain technology," in *Proc. IEEE 18th Int. Conf. High Perform. Comput. Commun., IEEE 14th Int. Conf. Smart City, IEEE 2nd Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS)*, Dec. 2016, pp. 1392–1393.

[47] S. Ibba, A. Pinna, M. Seu, and F. E. Pani, "CitySense: Blockchain-oriented smart cities," in *Proc. XP Scientific Workshops*, 2017, pp. 1–5.

[48] C. Oham, S. S. Kanhere, R. Jurdak, and S. Jha, "A blockchain based liability attribution framework for autonomous vehicles," 2018, *arXiv:1802.05050*. [Online]. Available: http://arxiv.org/abs/1802.05050

[49] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 2663–2668.

[50] B. Leiding and W. V. Vorobev. (2018). *Enabling the Vehicle Economy Using a Blockchain-Based Value Transaction Layer Protocol for Vehicular Ad-Hoc Networks*. [Online]. Available: https://www.chorus.mobi/

[51] P. K. Sharma, S. Y. Moon, and J. H. Park, "Block-VN: A distributed blockchain based vehicular network architecture in smart city," *J. Inf. Process. Syst.*, vol. 13, no. 1, pp. 184–195, 2017.

[52] V. Ortega, F. Bouchmal, and J. F. Monserrat, "Trusted 5G vehicular networks: Blockchains and content-centric networking," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 121–127, Jun. 2018.

[53] H. Khelifi, S. Luo, B. Nour, H. Moungla, and S. Hassan Ahmed, "Reputation-based blockchain for secure NDN caching in vehicular networks," in *Proc. IEEE Conf. Standards Commun. Netw. (CSCN)*, Oct. 2018, pp. 1–6.

[54] H. Shafagh, L. Burkhalter, A. Hithnawi, and S. Duquennoy, "Towards blockchain-based auditable storage and sharing of IoT data," in *Proc. Cloud Comput. Secur. Workshop (CCSW)*, 2017, pp. 45–50.

[55] K. R. Ozyilmaz, M. Dogan, and A. Yurdakul, "IDMoB: IoT data marketplace on blockchain," in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Jun. 2018, pp. 11–19.

[56] D. López and B. Farooq, "A multi-layered blockchain framework for smart mobility data-markets," *Transp. Res. C, Emerg. Technol.*, vol. 111, pp. 588–615, Feb. 2020.

[57] Y. Zhang, D. He, and K.-K.-R. Choo, "BaDS: Blockchain-based architecture for data sharing with ABS and CP-ABE in IoT," *Wireless Commun. Mobile Comput.*, vol. 2018, Nov. 2018, Art. no. 2783658.

[58] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2011, pp. 53–70.

[59] R. Radhakrishnan and B. Krishnamachari, "Streaming data payment protocol (SDPP) for the Internet of Things," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1679–1684.

[60] C. Mueller. (2018). *Volkswagen and IOTA Build the Future*. [Online]. Available: https://helloiota.com/volkswagen-and-iota-build-the-future

[61] R. Overko, R. H. Ordonez-Hurtado, S. Zhuk, P. Ferraro, A. Cullen, and R. Shorten, "Spatial positioning token (SPToken) for smart mobility," 2019, *arXiv:1905.07681*. [Online]. Available: http://arxiv.org/abs/1905.07681

[62] P. C. Bartolomeu, E. Vieira, and J. Ferreira, "IOTA feasibility and perspectives for enabling vehicular applications," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–7.
[63] S. K. Pinjala and K. M. Sivalingam, "DCACI: A decentralized lightweight capability based access control framework using IOTA for Internet of Things," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Apr. 2019, pp. 13–18.
[64] (2020). *Infura: Secure and Scalable Access to Ethereum Apis and IPFS Gateways*. [Online]. Available: https://infura.io/
[65] C.-I. Fan, Y.-F. Tseng, H.-P. Su, R.-H. Hsu, and H. Kikuchi, "Secure hierarchical bitcoin wallet scheme against privilege escalation attacks," in *Proc. IEEE Conf. Dependable Secure Comput. (DSC)*, Dec. 2018, pp. 1–8.
[66] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," 2015, *arXiv:1506.03471*. [Online]. Available: http://arxiv.org/abs/1506.03471
[67] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," 2017, *arXiv:1708.03778*. [Online]. Available: http://arxiv.org/abs/1708.03778
[68] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.
[69] D. Nunez. (2019). *Umbral: A Threshold Proxy Re-Encryption Scheme*. [Online]. Available: https://raw.githubusercontent.com/nucypher/umbral-doc/master/umbral-doc.pdf
[70] M. Egorov, M. Wilkison, and D. Nunez, "NuCypher KMS: Decentralized key management system," 2017, *arXiv:1707.06140*. [Online]. Available: http://arxiv.org/abs/1707.06140
[71] (Feb. 2020). *Parity Secret Store*. [Online]. Available: https://wiki.parity.io/Secret-Store
[72] C. Prandi, S. Mirri, S. Ferretti, and P. Salomoni, "On the need of trustworthy sensing and crowdsourcing for urban accessibility in smart city," *ACM Trans. Internet Technol.*, vol. 18, no. 1, pp. 1–21, Dec. 2017.
[73] H. C. Pohls, "JSON sensor signatures (JSS): End-to-End integrity protection from constrained device to IoT application," in *Proc. 9th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, Jul. 2015, pp. 306–312.
[74] S. Migliorini, "Enhancing blockchain smart-contracts with proof-of-location," in *10th Int. Conf. Geographic Inf. Sci.*, 2018, pp. 1–8.
[75] N. Kumar, R. Iqbal, S. Misra, and J. J. P. C. Rodrigues, "An intelligent approach for building a secure decentralized public key infrastructure in VANET," *J. Comput. Syst. Sci.*, vol. 81, no. 6, pp. 1042–1058, 2015.
[76] Foamspace Corp. *FOAM Whitepaper*. (2018). [Online]. Available: https://foam.space/publicAssets/FOAM_Whitepaper.pdf
[77] L. Wolberger and V. Fedyukovych. (2018). *Zero Knowledge Proof of Location*. [Online]. Available: https://platin.io/yellowpaper
[78] M. Zichichi, M. Contu, S. Ferretti, and V. Rodríguez-Doncel, "Ensuring personal data anonymity in data marketplaces through sensing-as-a-service and distributed ledger," in *Proc. 3rd Distrib. Ledger Technol. Workshop Co-Located*, Ancona, Italy, Feb. 2020, pp. 1–16.
[79] S. Distefano, G. Merlino, and A. Puliafito, "Sensing and actuation as a service: A new development for clouds," in *Proc. IEEE 11th Int. Symp. Netw. Comput. Appl.*, Aug. 2012, pp. 272–275.
[80] M. Alarbi and H. Lutfiyya, "Sensing as a service middleware architecture," in *Proc. IEEE 6th Int. Conf. Future Internet Things Cloud (FiCloud)*, Aug. 2018, pp. 399–406.
[81] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by Internet of Things," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 1, pp. 81–93, 2014.
[82] C. Jentzsch. (Nov. 2016). *Decentralized Autonomous Organization to Automate Governance*. [Online]. Available: https://lawofthelevel.lexblogplatformthree.com/wp-content/uploads/sites/187/2017/07/WhitePaper-1.pdf
[83] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: K-anonymity and its enforcement through generalization and suppression," SRI Int., Menlo Park, CA, USA, Tech. Rep. SRI-CSL-98-04, 1998.
[84] C. Dwork, "Differential privacy," in *Proc. Encyclopedia Cryptography Secur.*, 2011, pp. 338–340.
[85] D. Dias and L. H. M. K. Costa. (2018). *CRAWDAD Dataset Coppe-Ufrj/Riobuses (V. 2018-03-19)*. [Online]. Available: https://crawdad.org/coppe-ufrj/RioBuses/20180319
[86] (2020). *Skynet: Build a free Internet*. [Online]. Available: https://support.siasky.net/article/bbpwa0onbq-how-long-will-my-data-stay-online
[87] (Oct. 2019). *IOTA Nodes*. [Online]. Available: https://iota-nodes.net/
[88] V. Jacobson, "Congestion avoidance and control," in *Proc. Symp. Proc. Commun. Archit. Protocols*, 1988, pp. 314–329.
[89] (2020). *SIA: Uploading Data*. [Online]. Available: https://support.siasky.net/article/xddv04da2r-uploading-to-sia
[90] (Oct. 2019). *Dataset and Scripts Github Repository*. [Online]. Available: https://github.com/miker83z/testingIOTA
[91] (2019). *Radix knowledge base*. [Online]. Available: https://docs.radixdlt.com/kb/
[92] J. Poon and V. Buterin. (2017). *Plasma: Scalable Autonomous Smart Contracts*. [Online]. Available: https://www.plasma.io/plasma-contracts.html

MIRKO ZICHICHI received the bachelor's degree *(summa cum laude)* in computer science from the University of Palermo, in 2017, and the master's degree *(summa cum laude)* in information science for management from the University of Bologna, in 2019.

He joined the Ontology Engineering Group (OEG), Universidad Politécnica de Madrid, in 2019. He is a Doctoral Researcher with the Law, Science, and Technology Joint Doctorate—Rights of Internet of Everything, funded by Marie Skłodowska-Curie Actions. His Ph.D. research focuses on the location privacy and inference in online social networks and on the use of distributed ledger technologies and smart contracts for the protection and distribution of individuals' personal data.

STEFANO FERRETTI (Member, IEEE) received the Laurea *(summa cum laude)* and Ph.D. degree in computer science from the University of Bologna, in 2001 and 2005, respectively. He was an Associate Professor with the Department of Computer Science and Engineering, University of Bologna. He has been an Associate Professor with the Department of Pure and Applied Sciences, University of Urbino "Carlo Bo," since 2020. His current research interests include distributed systems, complex networks, data science, fintech and blockchain technologies, multimedia communications, hybrid, and distributed simulation. He is on the Editorial Board of the *Simulation Modelling Practice and Theory* (SIMPAT) journal (Elsevier) and the *Encyclopedia of Computer Graphics and Games* (Springer). He is on the Technical Committee of *Computer Communications* (Elsevier) and *Online Social Networks and Media* (Elsevier). He was an Editor of special issues on other international journals, such as CPE (Wiley) and ComCom (Elsevier). He acted as the chair for several conferences and workshops within flagship conferences, e.g., ACM Mobisys and IEEE InfoCom.

GABRIELE D'ANGELO (Member, IEEE) received the Laurea *(summa cum laude)* and Ph.D. degree in computer science from the University of Bologna, Italy, in 2001 and 2005, respectively. He is an Assistant Professor with the Department of Computer Science and Engineering, University of Bologna. His research interests include parallel and distributed simulation, distributed systems, online gaming, and computer security. He has been a Technical Program Committee Member of INFOCOM since 2019. Since 2011, he has been on the Editorial Board of the *Simulation Modelling Practice and Theory* (SIMPAT) journal (Elsevier).