LA-MQTT
: Location-aware publish-subscribe communications for the Internet of Things

LA-MQTT: Location-aware publish-subscribe communications for the Internet of Things

FEDERICO MONTORI, University of Bologna, Italy
LORENZO GIGLI, University of Bologna, Italy
LUCA SCIULLO, University of Bologna, Italy
MARCO DI FELICE, University of Bologna, Italy

Nowadays, several Internet of Things (IoT) deployments use publish-subscribe paradigms to disseminate IoT data to a pool of interested consumers. At the moment, the most widespread standard for such scenarios is MQTT. We also register an increasing interest in IoT-enabled Location-Based Services, where data must be disseminated over a target area and its spatial relevance as well as the current positions of the consumers must be taken into account. Unfortunately, the MQTT protocol does not support location-awareness, hence it may result in notifying consumers that are geographically far from the data source, causing increased network overhead and poor Quality of Service (QoS). We address the issue by proposing LA-MQTT, an extension to standard MQTT supporting spatial-aware publish-subscribe communications on IoT scenarios. LA-MQTT is broker-agnostic and fully backward compatible with standard MQTT. As monitoring the position of subscribers over time may cause privacy concerns, LA-MQTT carefully supports location privacy preservation, for which the optimal trade-off with the QoS of the spatial notifications is addressed via a learning-based algorithm. We demonstrate the effectiveness of LA-MQTT by experimentally evaluating its features via large-scale hybrid simulations, including real and virtual components. Finally, we provide a Proof of Concept real implementation of a LA-MQTT scenario.

CCS Concepts: • **Networks** → **Location based services**; *Middle boxes / network appliances*; • **Security and privacy** → **Privacy-preserving protocols**; *Pseudonymity, anonymity and untraceability*.

Additional Key Words and Phrases: MQTT, Location Based Services, Reinforcement Learning, Privacy

1 INTRODUCTION

The Internet of Things (IoT) is now a primary element in our everyday life and connects billions of heterogeneous devices together to form machine- and human-driven interactions [2]. Indeed, predictions for years to come confirm a steep growth of devices connected to the internet, estimating around 30 billion devices with an IP address by 2023, which is more than three times the world's population [13]. Many industrial, social, and public contexts benefit from this growth, as the need for instantaneous data is fulfilled more and more on a real-time basis, rather than having to rely on historical reports. As, now, data can be generated from any kind of device, from wearables to industrial installments, a rapid deployment of off-the-shelf services that are relevant to the potential consumer is of paramount importance [49][35]. An important criterion for relevance is location, in fact, several IoT-based services are strongly context-aware in terms of location suitability for consumers: examples include IoT-enabled parking systems [1], mobile crowdsensing [39], and location-aware alerting and notification systems in indoor environments such as residential buildings [41] or museums [3], just to name a few. These are called Location-Based Services

Authors' addresses: Federico Montori, federico.montori2@unibo.it, University of Bologna, Bologna, Italy; Lorenzo Gigli, lorenzo.gigli@unibo.it, University of Bologna, Bologna, Italy; Luca Sciullo, luca.scullo@unibo.it, University of Bologna, Bologna, Italy; Marco Di Felice, marco.difelice3@unibo.it, University of Bologna, Bologna, Italy.

2577-6207/2022/4-ART $15.00
https://doi.org/10.1145/3529978

(LBS) and are a major pillar in the context of Smart Cities of the future [46].

Because IoT data is typically stored in a common point of access, such as a Cloud, it is indispensable to provide on-demand access means to potential clients via a configurable and flexible approach which ensures that data reaches the correct destination. A common paradigm that accomplishes these requirements for shared resources is the publish-subscribe, where common resources are updated by service producers and service consumers automatically get updates about the resources they subscribed to [44]. A service broker enables such a communication and acts as a middleware in such a scenario, by guaranteeing the complete temporal and communication decoupling between the data producers and the data consumers. Although various standards exist in such a direction, MQTT (Message Queuing Telemetry Transport) is by far the most used publish-subscribe application protocol for IoT. It is an OASIS and ISO standard since 2013 [23] and it has been used in a variety of IoT-based contexts [32]. MQTT is lightweight, which makes it suitable for machine-to-machine ecosystems where messages are tiny and computational capabilities are limited, and it includes Quality of Service (QoS) and reliability mechanisms to guarantee message delivery. At the same time, some features of MQTT make it hardly suitable for wide and novel IoT deployments, such as IoT-based LBS. Indeed, in its original state, the topic string is the only mechanism offered by the protocol to filter the relevant messages on the broker and to disseminate them to the subscribers. Vice versa, in location-aware IoT applications, the service providers are the LBSs, therefore they are geolocated and the information they produce is likely to be relevant only in their neighborhood. If the MQTT broker covers a wide area, this may introduce significant network overheads and affect the QoS experienced by service consumers, since they may get data that is spatially irrelevant to them, even though the type of data (*i.e.* the MQTT topic) is exactly the one they subscribed to. As an example, let us think about a city-wide Smart Parking infrastructure, where each parking lot is equipped with a sensor that determines whether the spot is free or occupied. In such case, each sensor would broadcast their parking data to the MQTT broker, which would then notify every driver that subscribed to parking data. However, drivers that are moving far from the parking area will unlikely be interested to such information and, on the other hand, a massive amount of useless network traffic will be generated.

In order to cope with such issues, in this paper we propose `LA-MQTT` (Location-Aware MQTT), an extension to the traditional MQTT protocol for spatial-aware communications. `LA-MQTT` routes the data generated by service producers only to consumers for which such data is relevant in terms of both topic and location; in other words, it limits the amount of consumers to be notified to only the ones that are geographically inside the area of competence defined by the service producer. This is not the first scientific proposal towards location awareness in MQTT (e.g. [21]), however, our solution is both unique and complete, in that (*i*) it is the first broker-agnostic solution, and (*ii*) it also carefully considers the privacy implications of adding location awareness to a preexisting scheme. Regarding the first aspect, our solution is implemented around a common MQTT broker, without altering its implementation, while geo-processing capabilities are provided by a separate service relying on standard MQTT publish-subscribe facilities. Secondly, notifying users on top of their location involves somehow knowing their position. This is unavoidable, yet it causes privacy concerns, as user location is clearly a source of private information that, if acquired by malicious entities, can lead to harmful consequences. For this reason, the paper also proposes privacy-preserving schemes that are under the control of the clients and allows to capture the trade-off between QoS of the notifications and preservation of the client's location. More in details, three main scientific contributions are proposed in this study:

- We describe the software architecture and the implementation of `LA-MQTT`, as a broker-agnostic and MQTT-compliant solution to support the dissemination of messages on geofences of arbitrary shapes. The architecture includes a client library, running on the mobile IoT device and adding location-based publish-subscribe primitives, and a backend service providing the message geo-filtering functionalities.

- We design proper privacy preservation mechanisms integrated within the client library, and combining well-known techniques of GPS perturbation and dummy routes. In addition, we propose a Reinforcement Learning (RL)-based algorithm to tune the privacy parameters in an autonomic and adaptive way, so that each client is able to achieve the desired trade-off between Spatial Precision (SP) and Privacy Preservation (PP) metrics. The privacy preservation mechanism is designed as an integral part of LA-MQTT and allows for integration any other privacy-aware techniques in the framework. In this sense, GPS perturbation and dummy routes are used as specimens.
- We evaluate LA-MQTT through a twofold evaluation study. The effectiveness of the spatial filtering mechanism and the privacy solutions, including the convergence of the learning-based algorithm, are investigated through a hybrid simulation environment in which the real software agents of LA-MQTT are fed by simulated entities of Mobile Clients (MCs) on an urban scenario. Moreover, we provide a Proof-of-Concept (PoC) implementation in a small-testbed involving an IoT board as data producer and an Android application as data consumer, and analyze the average delay of the geo-dissemination process.

The paper is organized as follows. Section 2 outlines related works in the fields of MQTT extension and location-based privacy. Section 3 presents the system model, Section 4 describes in detail the software architecture and all the entities involved. Section 5 deals with privacy concerns, introduces our privacy-preserving solutions and presents the proposed reinforcement learning algorithm. Section 6 shows the evaluation results in a large-scale simulated deployment. Section 7 presents the PoC implementation and, finally, Section 8 concludes the paper.

2 RELATED WORK

This paper lays its foundations over three main pillars of research: the MQTT protocol, privacy preservation in Location-Based Services (LBS) and location awareness in publish-subscribe systems. In this section we will address the backgrounds and related literature that touch upon these three themes, with a particular focus on their merge points.

2.1 MQTT-related research

The MQTT (Message Queuing Telemetry Transport) is an IoT client-server publish-subscribe messaging protocol, proposed for the first time in 1999 and now OASIS and ISO standard [23]. It has found a wide applicability in constrained environments such as for communication in Machine to Machine (M2M) and IoT contexts where a small code footprint is required and/or network bandwidth is at a premium [21]; this is thanks to its features of being lightweight, open, easy to implement. Version 5 has been released in 2018; by default it uses IANA registered port number 8883 for SSL/TSL connections and 1883 port number for Non-TLS connections [23]. MQTT offers three different Quality of Service (QoS) for message delivery: “at most once”, when it does not matter if the connection to the application is interrupted for a while and message loss is tolerated, “at least once”, where message loss is not an option and duplicates are instead tolerated to cope with it, and “exactly once”, where both message loss and redundancy may lead to severe faults. A typical MQTT scenario features three main components:

(1) The MQTT Consumers (or Subscribers).
(2) The MQTT Producers (or Publishers).
(3) The MQTT Broker (or Server).

Both Consumers and Producers are classified as MQTT Clients and they can issue requests to the MQTT Server once the connection is established: such requests include publishing messages that may be of interest to other clients (carried out by the producers) and subscribing for updates (carried out by the consumers). The published messages are delivered to the consumers based on the MQTT Topic they refer to; *i.e.* if the consumer subscribes to a certain topic, it will receive only the updates, published by the producers, that pertain to such topic. The message exchange is mediated by the MQTT Broker, which accepts connections from the clients and keeps track

of active subscriptions, in order to deliver messages to the interested recipients upon their publish [23]. MQTT is indubitably one of the best and most widespread application protocols for IoT applications. It has several advantages such as being lightweight and energy efficient, granting many-to-many communication and relieving IoT devices from instantiating a continuously running service [32]. A multitude of MQTT Broker and libraries implementations have been proposed and the protocol itself is being actively discussed in the research community [32]. Performance comparison of MQTT against other IoT messaging protocols like CoAP or AMQP has been discussed, among others, in [31], [29] and [45]. Similarly, some protocol extensions have been proposed to tackle real-time communications [17] or the current lack of security, although enabling encryption as a separate feature causes performance overhead [15]. The security problem has been dealt with extensively in literature. One such solution was proposed in [30], where authors envision a security add-on to legacy MQTT to provide authorization, authentication and cryptography. This work builds on preexisting MQTT deployments and includes special actors (such as the Revocation Authority) in order to make it fully backward compatible.

2.2 Privacy in Location-Based Services

In order to maximize the QoS of Location-Based Services (LBS), it is usually required to know the position of potential clients a priori, as to match each of them with the most suitable and spatially relevant service. However, this obviously causes privacy concerns as the service provider would be aware at any time of where the users are and eventually be able to discover their habits and other sensitive information. As a matter of fact, privacy in mobile scenarios is a long-studied problem [7], with an incredible amount of works suitable for different scenarios. However, we restrict our research on geographical information and online and distributed privacy countermeasures. More in detail, these imply mobile users to implement their privacy mechanism on the fly and on the edge, therefore independently from the full knowledge base [42]. This excludes most of the offline strategies such as k-anonymity [18] and differential privacy [5], which aim to minimize the distinguishability of an individual among a group, as such a group is unknown to the mobile user. In fact, such kind of optimization is performed either in a central aggregator or, in case the central aggregator is untrusted, a set of intermediate agents is typically envisioned. This poses a delay and additional steps in information sharing and, therefore, it is not viable in our scenarios. In LBS, a common online strategy is location cloaking [12], which blurs the location of the user to a cloaked area before being sent over. However, again, the majority of the research works assumes an infrastructured or a peer-to-peer environment where users have, at least, a partial knowledge of the position of other users [10][38]. This enables them to optimize the cloaking procedure by producing the smallest perturbation that yields a target anonymity. Much less research is conducted towards producing a per-user perturbation with no knowledge about the others and, most of the times, this involves contextual information [40]. Another class of applicable online privacy mechanisms is that based on dummies techniques. Instead of perturbing their true location, users produce a set of dummy locations to confuse a potential attacker [26]. Further studies have been focusing on filtering out improbable dummies by taking into account temporal reachability and direction similarity [28], by maximizing the query entropy [36] and by caching wireless access points to store historical queries, preventing the user from submitting the location-aware queries frequently [37]. In other cases it is preferable to do the opposite, *i.e* sending over only a subset of the information, provided that the internal correlation is below a certain threshold, as in [34].

2.3 Location in Publish-Subscribe Systems

The issue of scalability, which is not taken into account by the MQTT standard itself, calls out for a consistent reduction of the messages in scenarios where the density of producers and consumers is high. A key observation here is that clustering messages spatially both reduces the burden at the broker level and provides more fine-grained information to subscribers, potentially increasing the QoS. Many studies propose their own solution to dense scenarios via distributed approaches on top of legacy MQTT, while some others propose their own publish-subscribe

method to tackle the problem. The work in [24] discusses a distributed architecture that divides the geographical area hierarchically into squares and places a dedicated broker for each of them. Routing is achieved by gateways that redirect subscribers to the pertinent broker. As no communication is actually required between brokers, the solution also supports heterogeneity. However, while being efficient at deployment time, the solution cannot adapt to localized increases of demand at run-time and requires several brokers to be deployed. A similar work is proposed in [25], where the communication between brokers is required, as they are organized in clusters. This allows to accommodate clients that connect to a broker that is not responsible for handling their publish and subscription operations. The previous works can be considered location-aware in the way the authors address the optimal placement of the MQTT brokers; however, they do not consider location-awareness for the publish-subscribe communications, which is the main goal of this paper. The most similar work to ours is [8] (and the related Android implementation in [16]), where the authors propose MQTT-g, a location-aware extension of MQTT that handles location-based relevance of topics for subscribers by encoding such information in the unused fields of MQTT messages. Subscribers need to periodically notify the broker about their position and a dedicated matching function will notify them only if they are located in the area pertinent to the information published by the producer. A similar work has been proposed in [22], where an extension of MQTT – in particular MQTT5, which carries along several enhancements, including the acceptance of additional properties according to the client's specific needs – for geolocation is presented. This proposal relies on the definition of the GLP payload, a special payload in which information about the geographically-based subscriptions is encoded in JSON. The proposal is very detailed in the definition of such payload and the area encoding, however being it still an initial stage, it does not come with validation nor implementation yet. Although the goal of [8] and [22] is similar, both solutions are quite different from ours, since they require to extend the broker functionalities: this implies a different implementation for each broker as well as requiring it to be open-source, leaving out the heterogeneity factor. Conversely, the proposed LA-MQTT does not manipulate the MQTT protocol by making use of unused bits and requiring a dedicated broker, instead it relies fully on legacy MQTT deployments by adding a component transparently. Also, the privacy for subscribers is not considered in [8] and [22]. MQTT has also been used as a location discoverer in more technology- and application-specific scenarios such as in [47], where a location-aware mechanism similar to ours is used to trigger BLE scan processes. Being it an application-specific approach, the actions to take on top of the location disseminated by the publishers is completely left to the subscribers, without any paradigm for generalized LBSs, nor the possibility of conveying arbitrary data together with the location. Besides these studies, other non-MQTT location-aware publish-subscribe solutions have been proposed. For instance, in [11] and [50] authors focus on data including text and geo-tagging (such as Twitter posts) and operate a spatial filtering on subscriptions based on grid cells. Subsequently, they rank the results according to their popularity score. These solutions, however, are not specifically made for IoT. Conversely, authors in [6] describe CUPUS, a publish-subscribe framework specifically tailored to Mobile Crowdsensing (MCS) scenarios, both to acquire data from sensors and to notify mobile users of nearby information of interest. In [19] authors propose Elaps, a publish-subscribe framework that aims to overcome the location deficiency of current solutions by delivering dedicated messages on top of consumer geofences attached to subscriptions. Similarly, GeoBroker, recently proposed in [20], is a novel Pub/Sub system that uses a geofence for both consumers and producers, in fact letting consumers decide how far data sources are meaningful to them and vice-versa. A similar solution is adopted in [9], where, again, both producers and consumers can specify their areas of influence and interest, respectively. Nevertheless, none of these solutions is compatible with existing standard deployments, plus they do not consider location privacy. Table 1 compares the major features of location-aware publish-subscribe solutions explored in this Section. More in detail, we report (i) whether the solution uses the MQTT standard or not (MQTT Compliance), (ii) whether it is fully compatible with existing brokers or it requires modifications to the broker implementation (Broker Compliance – requires MQTT Compliance), (iii) whether it uses the concept of geofencing [27] (Geofencing) and (iv) whether it does not strictly require multiple brokers to be deployed (Standalone).

Reference	MQTT Compliance	Broker Compliance	Geofencing	Standalone
Guo et al. [19]	✗	✗	✓	✓
Antonić et al. [6]	✗	✗	✗	✓
Chapuis et al. [9]	✗	✗	✓	✓
Venanzi et al. [47]	✓	✓	✗	✓
Bryce et al. [8]	✓	✗	✓	✓
Kawaguchi and Bandai [24]	✓	✗	✗	✗
Hasenburg and Bermbach [20]	✗	✗	✓	✓
LA-MQTT	✓	✓	✓	✓

Table 1. Review of related location-aware publish-subscribe solutions explored in Section 2.3.

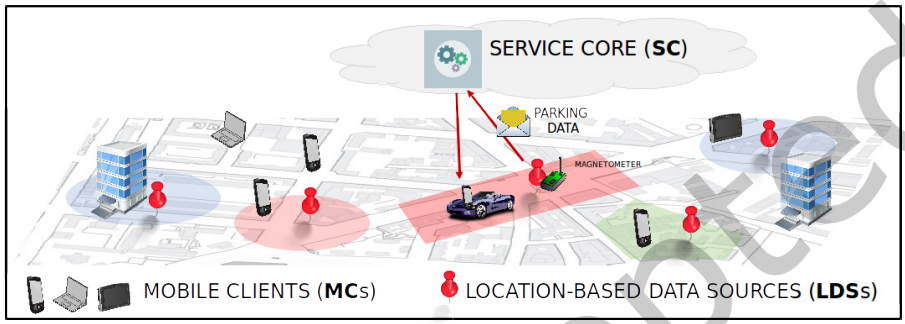


Fig. 1. The reference IoT scenario, with the actors involved (MCs, LDSs and the SC).

3 SYSTEM MODEL

Without loss of generality, our study addresses the generic IoT scenario depicted in Figure 1 and composed of two main actors, *i.e.* the Mobile IoT Clients (MCs), which play the role of service/data consumers and the Location-based Data Sources (LDSs), which play the role of service/data providers. More in detail, MCs are smartphones or IoT devices embedding a GPS sensor. Let M and N be the number of active MCs and LDSs, respectively. Let also $P_i = \langle lat_i, lon_i \rangle$ be the current position, in terms of latitude and longitude, of MC_i. Similarly to many IoT environments, we assume that the communication between the MCs and the LDSs occurs through a publish-subscribe paradigm, *i.e.* a MC registers its interest for a specific data channel and it is notified each time a new data instance belonging to such channel is produced by any of the LDSs. Each channel is identified by a unique topic name (a string) describing the service type. Let T_i be the list of topics of interest for MC_i. However, differently from existing solutions, we consider spatial constraints for the dissemination of IoT data, *i.e.* the identification of target MCs takes into account the topics of interest to which they subscribed *as well as* their current locations. An example of applicability can be the smart mobility application mentioned in the Introduction, through which a driver receives real-time notifications about spare parking spots in his whereabouts. To support this feature, we assume that an LDS_s is associated with a topic t_s that specifies both the data type (*e.g.* parking data) and a region of interest – defined here as a "geofence" – g_s, which defines the geographic region over which its data is of interest and must be delivered. Despite the common understanding about a geofence [33], we do not introduce any assumption regarding its shape, which can be circular or polygonal according to the LDS administrators' preferences. Let us define c_s to be the content to be delivered relative to t_s. In particular, LDS_s generates a new content c_s every f_s seconds. In Figure 1, the static magnetometer is part of LDS_{park} that notifies MCs that are both subscribed to its topic t_{park} and located within the rectangular geofence g_{park} about the presence of a spare parking

spot. According to our notation above, LDS_{park} produces the availability data c_{park} of the parking spot every f_{park} seconds. The communication between MUs and LDSs occurs through a third-party infrastructure generically called as the Service Core (SC); thanks to the logical decoupling between the data producers and the data consumers offered by the SC, the MCs can focus on the topic of interests rather than on the network addresses of the data sources. However, to implement the publish-subscribe communication, the MC must periodically notify the remote SC about its own GPS position P_i. Let f_i be the time interval between two consecutive GPS publishing events for MC_i. We then define a message generated by LDS_s to be *relevant* for MC_i whether: (*i*) $t_s \in T_i$, *i.e.* the topic of the LDS_s is of interest for MC_i; (*ii*) $P_i \sqsubset g_s$, *i.e.* MC_i is within the geofence region of LDS_s, where the operator $\sqsubset$ is used here to denote the spatial inclusion of a 2D point within a 2D region.

Moreover, since MCs are continuously sharing their GPS position with the SC, they might be exposed to privacy leaks, as the SC might be able to associate a user identity to its current location and/or to track the trajectories of a user over time. For this reason, it is necessary to investigate privacy mechanisms through which a MC is able to alter its GPS position before transmitting it to the SC, without causing a major performance drawback. Let $\tilde{P}_i$ be the GPS position of MC_i estimated by the SC. We denote as *Privacy Preservation* of MC_i (PP_i in short) the average geographical distance between $\tilde{P}_i$ and P_i, *i.e.* between the real position of MC_i and the estimation performed by the SC. As in the vast majority of location-based scenarios, the *PP* metric is in a trade-off with the accurate performance of the system [4]. Here, we represent such accuracy with the Spatial Precision metric (SP_i in brief), defined as the ratio of relevant notifications over the total ones received by MC_i. The *SP* metric can be considered as a proxy for the quality of data as perceived by the MC. Clearly, the higher the PP_i, the lower the SP_i. The aim of the study is then to design a proper location-aware publish-subscribe mechanism able to efficiently address the aforementioned trade-off.

4 LA-MQTT: SOFTWARE ARCHITECTURE

In order to tackle the challenge presented in the previous section, we propose a new publish-subscribe protocol called LA-MQTT, a seamless extension of the popular MQTT protocol that aims at supporting location-aware IoT communications. Specifically, LA-MQTT has been designed according to these principles: (*i*) *standard compatibility*, *i.e.* no changes to message formats of the MQTT protocol have been introduced; (*ii*) *broker compatibility*, *i.e.* LA-MQTT is implemented as an external add-on and hence does not require any specific support from the broker; (*iii*) *fine-grained geofencing* support, *i.e.* our architecture allows to define geofence regions of any shape and not only circular as in [8]; (*iv*) *privacy-awareness*, *i.e.* we propose client-side mechanisms and algorithms aimed at guaranteeing the best trade-off between the *SP* and *PP* metrics mentioned in the previous section. The LA-MQTT software architecture is shown in Figure 2. It includes three main components: a legacy MQTT broker, the LA-MQTT client library, and the LA-MQTT backend. Regarding the first, we tested the operations of LA-MQTT with Eclipse Mosquitto[1], due to its widespread usage among the IoT research communities and its open-source license; however, we remark that LA-MQTT is broker-agnostic, and hence it can be seamlessly integrated with other tools available on the market. The LA-MQTT client library is a tiny software layer that supports geo-communication facilities for both MCs and LDSs, such as publishing a new geofence (for the LDSs) or subscribing to it (for the MCs). The backend is in charge of processing the LA-MQTT messages and triggering spatial notifications associated to an entry event within a specific geofence. It may be deployed on the same host of the MQTT broker, or on a different host. In the following, we detail the internal structure and the operations of each component. Figure 2 is also described in detail with the help of an example in Section 4.3.

[1]Eclipse Mosquitto, an open-source MQTT broker (https://mosquitto.org/)

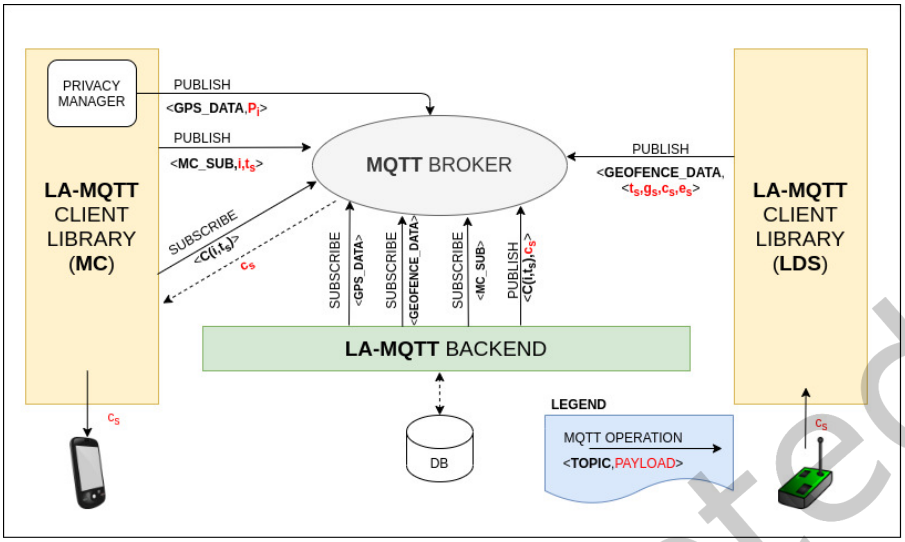


Fig. 2. The LA-MQTT software architecture [figure best viewed in colors].

API	Subject	MQTT OP	Topic	Payload
Position publish	MC	Publish	GPS_DATA	{position: P_i}
Topic subscription	MC MC	Subscribe Publish	$C(i, t_s)$ MC_SUB	* { mc: i, topic: t_s }
Geofence publish	LDS	Publish	GEOFENCE_DATA	{topic: t_s, content: c_s, region: g_s, event: e_s}
Content publish	Backend	Publish	$C(i, t_s)$	{content: c_s}

Table 2. The LA-MQTT publish-subscribe operations.

4.1 The LA-MQTT client library

The LA-MQTT client library is used by both MC and LDS devices, as it enables location-aware publish-subscribe operations against the remote MQTT broker. The underlying data transfer is managed by the legacy MQTT protocol: no changes to the MQTT messages' formats or to the messages' sequence have been made. However, the LA-MQTT client library introduces a new API for location-based communications over the MQTT publish-subscribe primitives. This has been implemented by defining, for the LA-MQTT messages only, the following: (*i*) a reference naming convention for the MQTT topics and (*ii*) a uniform payload structure, encoded through the JSON language. Table 2 summarizes the list of topic names and payload formats for each of the three main functionalities offered by the LA-MQTT client library:

- *Position Publish*. The API allows the MC to transmit its current GPS position to the MQTT broker. The operation is mapped to a basic MQTT publish action, where the topic is constrained to a predefined string (GPS_DATA), and the latitude and longitude values are embedded within the payload. Before performing this action, the client activates the privacy mechanisms described later in Section 5.
- *Topic Subscription*. The API allows the MC to subscribe to a topic. As a result, the MC is notified each time any LDS generates a new content of that topic, provided that the location constraints are met. The operation is inherently mapped to two actions. The first one is basic MQTT subscribe action: the topic of interest is built as a concatenation of the topic provided by the user, an identifier of the MC, and a predefined prefix. In this

way, LA-MQTT manages private subscriptions for each MC and topic of interest; this choice is justified by the need of providing filtered dissemination of IoT messages towards those MCs that meet the location-based requirements explained later in this Section. We indicate with $C(i, t_s)$ the newly created topic for MC_i and LDS_s. The second action, beside the MQTT subscription, is a MQTT publish request issued by MC_i. Such a request is issued with default topic MC_SUB and payload $\langle t_s, i\rangle$: the purpose is to notify the backend about the subscription performed by MC_i, as better detailed in Section 4.2.

- *Geofence Publish*. The API allows the LDS to publish messages that must be delivered to all the interested MCs located within a target geofence. The MQTT topic is constrained to a predefined string (GEOFENCE_DATA), while the payload is composed of four fields: the LDS topic t_s, the advertised content c_s, the target geofence g_s, and a mobility event e_s. The advertised content is the application-dependent content (c_s) that must be ultimately received by the MCs (*e.g.* parking spot availability report with respect to the example in Figure 1, including also the position of the parking spot). The geofence g_s must be encoded through the GEOJSON RFC 7946 format[2][3]. Even though we claimed that a topic is relevantfor a MC when the latter is located *inside* the relative geofence, we may envision more complex spatial relation. To reflect this, we add the mobility event e_s, that captures the condition under which the notification must be fired. At present, two basic options are supported, $e_s \in \{\texttt{IN} \mid \texttt{OUT}\}$, according to whether the MC is inside or outside the geofence region. We plan to extend the list of mobility events as future work, for instance including the event of dwelling in the geofence for more than a certain time period.

4.2 The Backend library

The LA-MQTT backend is executed as a background process either on the same host of the MQTT broker or on a separate host. At the system startup, the backend library subscribes to the GPS_DATA topic: hence, it is notified every time a new GPS position update is sent by any MC_i. The entry $\langle i, P_i\rangle$ is stored on a MongoDB database. Similarly, it subscribes to the GEOFENCE_DATA topic, thus it is also notified every time a new message is sent by any LDS_s. In this case the entry $\langle t_s, g_s, c_s, e_s\rangle$ is stored on a MongoDB database. To determine the list of potential receivers for each message generated by LDS_s, the backend needs to know T_i, *i.e.* the list of subscriptions of each MC_i; however, the MQTT broker might not expose such information through an API. For this reason, the backend subscribes to the MC_SUB channel, being notified each time MC_i registers to topic t_s. The backend includes a geoprocessing module that is activated every time a new GPS_DATA is published by any MC_i and verifies whether the event spatial constraint is satisfied (*e.g.* $P_i \sqsubset g_s$ if e_s=IN) and $t_s \in T_i$. If both conditions are met, the backend publishes a new message on the private MQTT topic $C(i, t_s)$, setting c_s as MQTT payload. In addition, a similar process is executed when a LDS_s publishes a new content c_s: in this case, the module loops over all the MC_j with $0 < j \leq M$, as they might miss the notification otherwise, and publishes a new message with payload c_s on all the channels $C(j, t_s)$ for which $t_s \in T_j$ and the e_s condition holds. Finally, we notice that a MC might move within a geofence g_s, however it should be notified only once for each content c_s produced by LDS_s. For this reason, the backend includes a caching module that keeps track of the history of notifications delivered by LDS_s to MC_i; this is implemented by storing (*i*) a sequence number N_s that counts the GEOFENCE_DATA messages produced by LDS_s, and (*ii*) a cache data structure $\langle N_s(i), i\rangle$, where $N_s(i)$ is the sequence number of the latest message produced by LDS_s that was delivered to MC_i. When processing a GPS update from MC_i, the module verifies that $N_s(i) < N_s$; only in that case, provided that $t_s \in T_i$ and $P_i \sqsubset g_s$, the message is published on the MQTT broker, with topic $C(i, t_s)$ and payload c_s.

[2] https://geojson.org/

[3] Polygon geometry is the only supported at present, for circles we needed to add a custom encoding.

4.3 Workflow Example

In order to better ease the understanding of the workflow presented in Figure 2, we go over the whole procedure with the aid of an exemplary scenario. In particular, we make use of the smart parking example, also mentioned in Figure 1. The scenario is under the control of a deployed Legacy MQTT Broker, together with the `LA-MQTT` backend. The latter performs three subscriptions at deployment stage: one for `GEOFENCE_DATA` (coming from LDSs), one for `GPS_DATA` and one for `MC_SUB` (both coming from MCs). Now, let us assume to deploy a smart parking system, in which each parking spot is an independent LDS equipped with the `LA-MQTT` client library, while each driver looking for a parking spot is a MC, also equipped with the `LA-MQTT` client library. As soon as the MC starts looking for a parking spot, it performs two operations. First it subscribes to the topic $C(i, t_s)$, obtained by concatenating its unique id i and the generic topic "Parking" t_s. Second, it publishes its interest in parking spots using the topic `MC_SUB` and i and t_s as a payload. This is captured by the `LA-MQTT` backend, which is then aware that MC_i is looking for a parking spot. MC_i starts then to periodically advertise its GPS position through a publish operation with the topic `GPS_DATA`. Meanwhile, the LDS periodically broadcasts the availability of the parking spot through a publish operation with topic `GEOFENCE_DATA`. This contains the generic topic "Parking" t_s, an area of interest that physically surrounds the LDS (g_s) and the actual availability of the parking spot (c_s). Since we assume that the parking spot is of interest for whom is located within the area of interest, we set e_s to `IN`. Both `GPS_DATA` and `GEOFENCE_DATA` are captured by the backend, which checks if the position of MC_i from its `GPS_DATA` is located within the geofence contained in `GEOFENCE_DATA`. If so, then the backend forwards the notification to MC_i by publishing on the topic $C(i, t_s)$ the availability of the parking spot. This message gets received by MC_i only.

5 `LA-MQTT`: PRIVACY MANAGEMENT

The privacy module is implemented within the client library, as it only affects the MCs during the position publish operation. The module takes in input the real position P_i of MC_i, and produces in output a new position $\tilde{P}_i$, generated in a way to address the *PP-SP* trade-off introduced in Section 3. In Section 2, we reviewed several privacy techniques for geolocation tracking applications. We note again that only a few of them fit the characteristics of our scenarios, such as the absence of cooperation among the MCs and the limited computational resources of some IoT/mobile devices. For this reason, we propose a modular, cost-effective, probabilistic algorithm that combines the two strategies explained below, by leveraging their advantages and hindering their drawbacks. It is worth mentioning that we do not force the choice of such strategies, as others could fit our framework. In this sense, they are used here with the purpose of validating the framework.

- *Strategy 1: Random Perturbation.* Let D indicate the total numbers of digits after the comma for a GPS latitude/longitude sample. Let $0 \leq d \leq D$ be the perturbation index. Given P_i, we generate $\tilde{P}_i$ by keeping the integer part of P_i and the first d digits of the real part; the remaining $D - d$ digits are replaced with random values. It is easy to see that the corner case $d = D$ produces no perturbation. The d parameter controls then the amount of perturbation and, consequentially, the trade-off between the *PP* and *SP* metrics. In Section 6, we investigate the impact of such a parameter on the system performance.
- *Strategy 2: Dummy Updates.* Let us consider sequences of $b + 1$ consecutive position publish operations by MC_i. In this strategy, the MC publishes its true GPS position (P_i) in only one of such operations, while in the other b cases, a random perturbed position with index d is generated and published on the MQTT broker. The index of P_i within the sequence is also randomly determined for each new sequence. Let then $S_i^l = \{\tilde{P^l}_{i,1}, \tilde{P^l}_{i,2}, ..., \tilde{P^l}_{i,b+1}\}$ be the l-th sequence of dummy updates, where $\exists\, 1 \leq j \leq b+1$ such as $\tilde{P^l}_{i,j} = P_i$. We refine the *PP* metric by computing it over each sequence S_i^l, as follows:

$$PP_i(S_i^l) = \frac{\sum_{j=1}^{b+1} dist(\tilde{P^l}_{i,j}, P_i)}{b+1} \quad (1)$$

where $dist(\cdot,\cdot)$ is the function returning the geographic distance between two GPS locations. Large values of b adversely impact the system responsiveness since the MC may be notified with a certain delay (or even not be notified at all) when entering a geofence, depending on the publish frequency f_i. Conversely, we recall that the PP metric empirically tends to increase for larger values of b, as one of the $b+1$ distances at the numerator in Equation 1 refers to the real position itself, therefore is always 0. This is not a monotonic behavior, however it is reasonable to expect such an increase because it is unlikely that b would assume values high enough to compromise this assumption. For these reasons the optimal tuning of the b parameter is critical; in Section 6 we investigate its impact on the system performance.

It is clear that the two methods above, if considered individually, present significant drawbacks. The random perturbation mechanism introduces a GPS noise that is scenario-dependent: clearly, for $d = 0$, the PP metric is maximized; however, the consequential reduction of SP might be significant, and adversely impact the user's experience. The dummy updates mechanism blurs the average position of the MCs; however, the broker might still be able to distinguish the true GPS updates from the dummy ones, *e.g.* by discarding the updates that are too far from the previous ones. We face these issue by introducing spatial correlation on the dummy updates and also taking advantage of the random perturbation. To this end, let P_i^s be the true initial position of MC_i. We construct b different trajectories: the starting points are computed by applying b times a random perturbation with index d to P_i^s, while the destinations are randomly generated within the scenario. Let $\delta_i^l = dist(P_i^l - P_i^{l+1})$ be the spatial distance covered by MC_i between two consecutive positions publish. For each fake update, the $\tilde{P}_{i,j}^{l+1}$ value is built by applying a δ_i^l movement on the j-th trajectory, starting from the previous position $\tilde{P^l}_{i,j}$. The term l is thus defined as the progressive step of each trajectory and it is the same across all the trajectories as they are generated along the real one. Figure 3 shows the three stages of the proposed privacy algorithm in action. The system performance, as well as the ability to cope with the PP-SP trade-off introduced in Section 3, depends on the proper tuning of the d and b operational parameters for each MC_i, as investigated in details in Section 6.3. In turn, the optimal tuning is affected by scenario-specific conditions, such as shape of the geofence, frequency of content updates from the LDSs, scenario geography, and many others that are mostly unknown by the MCs. Vice versa, for realism purposes we assume that MCs have only local knowledge of the environment, without any sort of cooperation among them. Given these constraints, the only way for the MC to determine the proper configuration of the privacy parameters is by dynamically learning it from the environment. For this reason, we propose an online learning algorithm that is executed by each MC in isolation, and that allows to configure the d and b parameters so that the PP-SP trade-off is dynamically addressed.

5.1 Learning-based Parameter tuning algorithm

We propose the application of Reinforcement Learning (RL) techniques to the problem of LA-MQTT parameter tuning. In brief, the RL framework addresses a wide category of situations where an agent must determine the optimal policies (*i.e.* sequences of actions) to complete a given task; to this aim, the agent interacts with the environment and receives a numeric reward after each action [43]. More formally, the RL framework can be represented as a Markov Discrete Process (MDP) $\langle S, A, R, TR\rangle$ where S is the set of States, A is the set of Actions, $R : \{S, A\} \rightarrow \mathbb{R}$ is the Reward function, expressing a numeric reward received by the agent when executing action $a_j \in A$ in state $s_i \in S$, and $TR : \{S, A\} \rightarrow S$ is the transition function, expressing the next state s_j after performing action a_k from state s_i (a deterministic environment is assumed). The goal of the RL agent is to determine the optimal policy function $\tau : S \rightarrow A$ that indicates the optimal action to execute at each state, so that the long-term reward is maximized. In our problem, the MDP is modeled as follows:

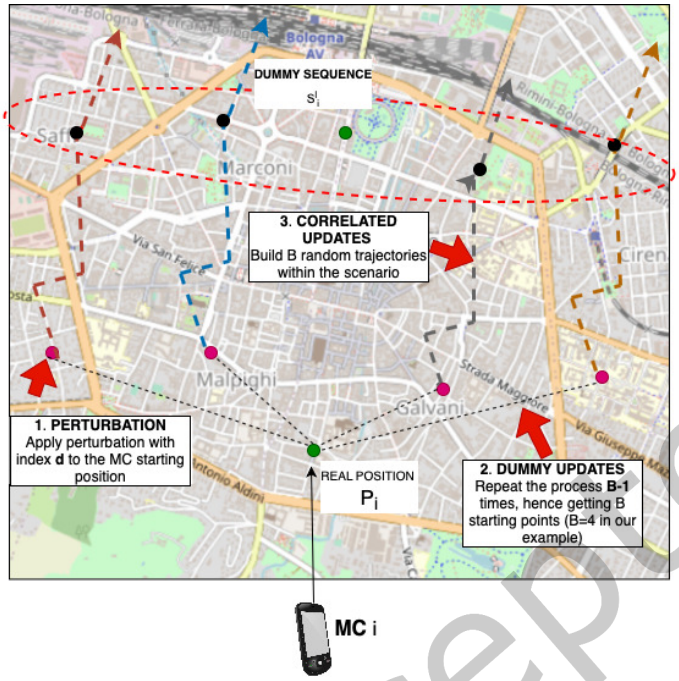


Fig. 3. The privacy mechanisms adopted by the `LA-MQTT` client.

- The set of States $S = \{s_1, s_2, ..., s_{|S|}\}$ describes the discrete position of the MC within the scenario. For this purpose, we assume a grid decomposition of the environment; let $|S|$ be the total number of cells, and $0 \leq s_j < |S|$ be the current cell containing the real position of MC_i.
- The set of Actions $A = \{a_1, a_2, ..., a_{|A|}\}$ describes the current privacy configuration of the MC. To this aim, let a_i be the couple $< b_j, d_k >$ where $0 \leq b_j < B_{max}$ is the current number of dummy updates and $1 \leq d_k < D$ is the current configuration of the perturbation index. It is easy to see that $|A| = D \cdot B_{max}$, where B_{max} is the maximum amount of dummy routes allowed.
- The Reward function R takes into account both the SP and PP metrics, and is computed as follows:

$$R_i = \gamma \cdot \bar{PP}_i + (1 - \gamma) \cdot SP_i \tag{2}$$

where γ is a parameter tuned by the MC, and controlling the wanted trade-off between PP and SP; for instance, by setting $\gamma = 1$, the system will maximally reward the actions that preserve the privacy, while for $\gamma = 0$ the system will give maximal priority to the quality of notifications received by the MC. In the previous Equation, the PP metric is normalized into the interval $[0 : 1]$, *i.e.*:

$$\bar{PP}_i = \left(\frac{PP_i}{PP_{max}} \right)^{\alpha} \tag{3}$$

where $0 < \alpha \leq 1$ is a parameter governing the sharpness of the curve – the lower the value of α, the faster $\bar{PP}_i$ increases for low values of PP_i –, and PP_{max} is a normalization factor.

For each state s_j and action a_k, each MC_i stores a Q-table that quantifies the average reward received by the agent while being in state s_j and executing action a_k. However, since the action is not correlated with the state update, we

consider a simplified version of the popular Q-learning algorithm [48], in which the Q-table at epoch t is updated as follows:

$$Q_i^t(s_j, a_k) = (1 - \eta) \cdot Q_i^{t-1}(s_j, a_k) + \eta \cdot R_i^t \tag{4}$$

where $0 \leq \eta \leq 1$ is a learning factor. Each Q-value is mapped into a probability value according to the well-known Boltzmann formula [43], which returns the probability of selecting action a_k when the MC_i is in state s_j:

$$PR_i^t(s_j, a_k) = \frac{e^{\frac{Q_i^t(s_j, a_k)}{T}}}{\sum_{a \in A} e^{\frac{Q_i^t(s_j, a)}{T}}} \tag{5}$$

where T is known as the temperature parameter, and controls the trade-off between exploration and exploitation phases.

Algorithm 1: The LA-MQTT RL-algorithm.

upon event Position Publish for MC_i**:** (every f_i intervals)
    if *new_step* **then**
        Set index of the true update $K = random(1, b + 1)$
        Update step number $l = l + 1$
        Reset counter within step $c = 0$
        Set *new_step* = false
    end
    if $K == c$ **then**
        Publish real GPS Position P_i
    else
        Publish dummy GPS Position $\tilde{P}_{i,c}^l$
    end
    Update counter within sequence $c = c + 1$
    if $c == b + 1$ **then**
        Set *new_step* = true
        update $\bar{PP}_i$ according to Equation 3
        Update the $\tilde{P}_i^{l+1}$ values on each of the B_{max} trajectories
    end

upon event MQTT notification received:
    Update SP_i metric for epoch t

upon event epoch t **completed:**
    get $s_j \in S$ from current position P_i
    update R_i according to Equation 2
    update $Q_i^t(s_j, a_k)$ according to Equation 4
    update $PR_i^t(s_j, a_k) \forall s_j \in S, a_k \in A$ according to Equation 5
    randomly select next action a^* to be executed based on $PR_i^t(s_j, a_k)$ values
    if $T > T_{min}$ **then**
        Decrease T by a ζ quantity according to Equation 6
    end
    Reset SP_i and PP_i metric values;

The pseudo-code of the overall procedure executed by the MCs is described by Algorithm 1, in which we assume the time to be divided into epochs of fixed length t_e. The algorithm is described as a reaction to three possible events: the position publish event, the reception of the MQTT notification and the completion of an epoch. Upon each position publish event, the MC checks whether it is about to generate a new step (line 2), as all the b dummy positions for step l, plus the real one, have been published. In such case, it determines randomly the index of the next true update among the $b + 1$ to be performed for step $l + 1$ (line 3). Then, the position publish takes place, by publishing the real one in case the index matches with the current iteration (line 9) or by publishing a pre-calculated position of the c-th dummy route (line 11). Once all the updates have been generated for step $l + 1$, the PP_i metric is updated (line 16) and the dummy positions for the next step are generated (line 17). Upon the reception of an MQTT update, MC_i updates its SP_i metric. Finally, at the end of each epoch, MC_i evaluates the reward function (Equation 2, line 25) and updates the Q-table and the $P_i^t(s_j, a_k)$ values $\forall s_j \in S$, $a_k \in A$ through Equations 4 and 5 (lines 26-27), respectively. In addition, the algorithm selects the action a^* to be executed during the next epoch: the action selection is performed probabilistically, according to the distribution given by Equation 5 (line 28). Finally, the T temperature, initialized with $T_{max} > 1$ is discounted by a fixed quantity ζ till reaching a minimum value $0 < T_{min} < 1$ (lines 29-31). The optimal tuning of the ζ parameter is critical since it determines the speed of moving from the exploration to the exploitation phase, in which the actions associated with higher Q-values are executed most of the time. In our system, we set ζ as follows:

$$\zeta = \frac{T_{max} - T_{min}}{|A| \cdot \Delta} \tag{6}$$

$|A|$ is the number of configurations to test, while $\Delta > 0$ indicates the average number of times in which a configuration should be selected. We evaluate the impact of Δ parameter on system performance in Section 6.

6 PERFORMANCE EVALUATION

In this section, we address scalability, spatial accuracy, and privacy management of `LA-MQTT` on a large-scale, immersive simulated environment. The characteristics of the latter are presented in Section 6.1. The geoprocessing capabilities of `LA-MQTT` are evaluated in Section 6.2. The impact of privacy parameters such as the number of dummy updates and perturbation index on the *SP*-*PP* metrics is shown in Section 6.3. Finally, in Section 6.4 we evaluate the performance of the learning-based parameter tuning algorithm described in Section 5.1.

The `LA-MQTT` Backend library, the `LA-MQTT` client library and the simulation environment used for evaluation in the present paper is open source and available online at `https://github.com/UniBO-PRISMLab/la-mqtt`.

6.1 Simulation environment

We consider the hybrid simulation environment shown in Figure 4 and composed of both real and virtual components. The real components are constituted by the elements of the `LA-MQTT` architecture previously presented, *i.e.* the client library and the backend, which are both coded in Typescript, and the MQTT broker, for which we used the Mosquitto tool. The virtual components are simulated MCs and LDSs entities which feed the `LA-MQTT` platform with synthetic data. More in detail, we deploy a tool to simulate the mobility of M MCs over an urban scenario; each MC moves according to a random waypoint model, with random speed in the range $[v_{min}, v_{max}]$. When reaching the destination, the MC stays still for an interval (again randomly selected between $[p_{min}, p_{max}]$) and then picks a new destination within the scenario. The custom mobility simulator was developed in Typescript. Each simulated MC publishes its GPS position on the MQTT Mosquitto broker via a private instance of the `LA-MQTT` client library at a constant, unique rate (*i.e.* $\forall i \in [1, M].f_i = f_{MC}$); in addition, each MC randomly subscribes to one topic among the T available ones. Similarly, we simulate the operations of N LDSs, randomly assigned to the T

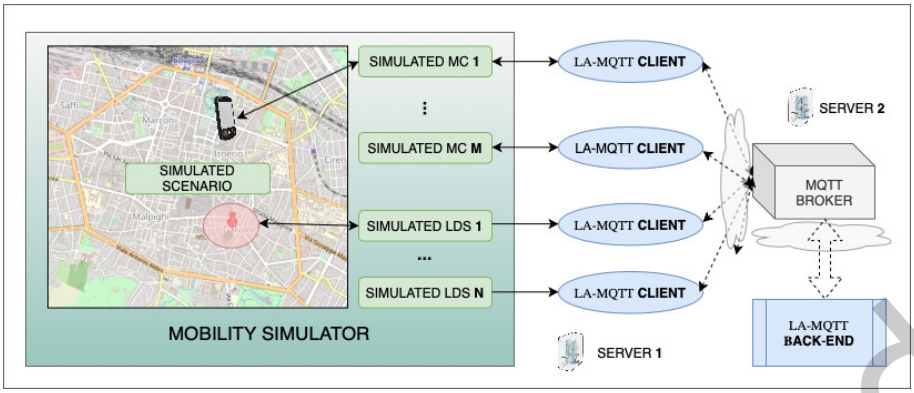


Fig. 4. The hybrid simulation environment.

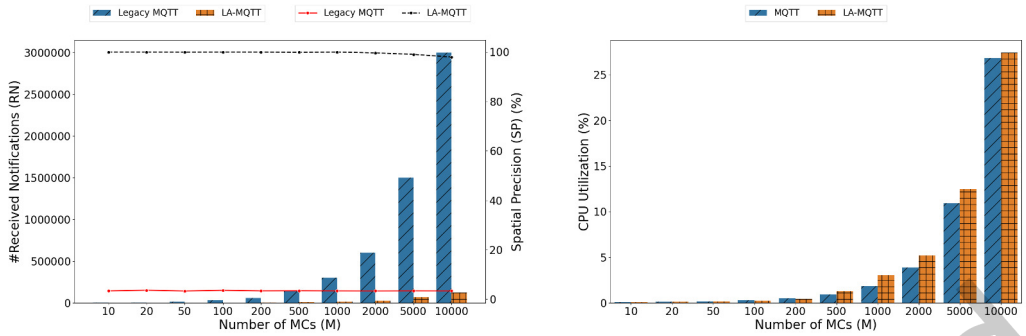
Parameter	Description	Value
M	Number of MCs	{10,20,50,100,200,500}
N	Number of LDSs	20
·	Scenario size	Bologna downtown (8,120km^2)
R	Geofence radius	{100,200,300,400,500,600}
T	Number of topics	3
f_{LDS}	Frequency of updates (from the LDSs)	60s
f_{MC}	Frequency of GPS sampling (from the MC)	10s
$[v_{min}, v_{max}]$	MC Min/max speed	[2, 5]m/s

Table 3. Default parameters' values used in the simulations.

topics: each LDS generates new data with the same constant frequency (*i.e.* $\forall s \in [1, N]. f_s = f_{LDS}$) and pertaining to one topic. We abstract from the physical meaning of the IoT data produced by each LDS; we will address the issue in Section 7, where an IoT use case with real-world sensor data is discussed. Although our solution supports geofences of any polygonal shape, the analysis is restricted to the case of circular geofences. In this way, we can investigate the system performance as a function of a single parameter, *i.e.* the geofence radius, denoted as R in the following and assumed equal for all the N LDSs. As for the MCs, each simulated LDS is connected to its private instance of the LA-MQTT client library, through which it can publish its data on the MQTT broker. The MQTT broker and the integrated simulation framework are installed on separate servers, communicating via a Gigabit Internet connection; as a result, the latency introduced by the communication channel, by the MQTT protocol and by the processing of spatial data are taken into account. The LA-MQTT backend library is installed on the same server hosting the simulation platform.
Unless stated otherwise, we simulate the LA-MQTT operations over the urban city center of Bologna (8,120km^2), by tuning the operational parameters as reported in Table 3. We consider real-time simulation: the duration of each run is equal to 30min. For each result shown in this section, we averaged the values over 5 runs.

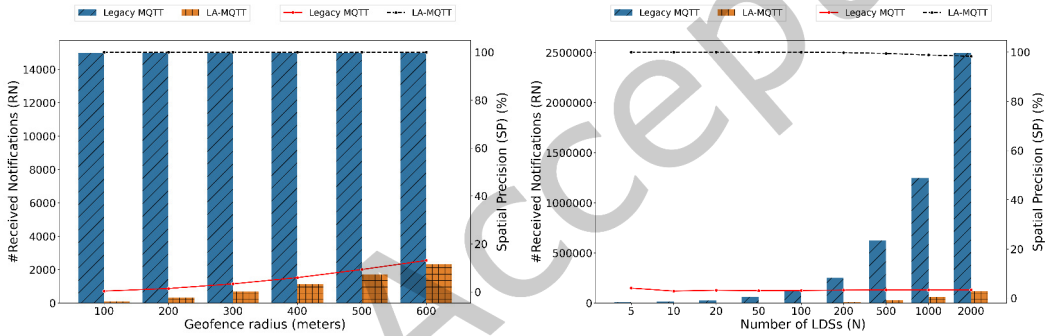
6.2 Geofiltering Analysis

As preliminary analysis, we investigate the ability of LA-MQTT to support location-based dissemination of IoT data over the scenario. Specifically, we focus on the performance of the geo-filtering mechanism, *i.e.* on the proper



(a) Varying the number of MCs (b) Avg CPU usage of the MQTT broker

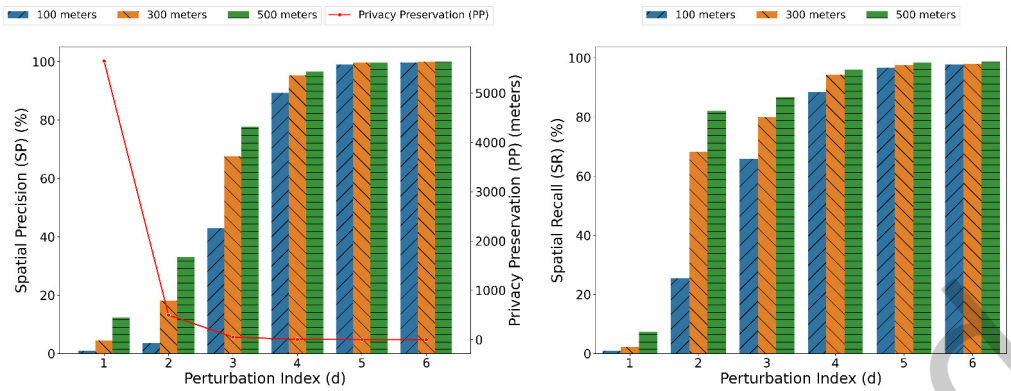
Fig. 5. The RN and SP metrics for the `LA-MQTT` and *Legacy MQTT* protocols when varying the number of MCs (M) are shown in Figure 5(a). The average CPU usage of the MQTT broker is shown in Figure 5(b).



(a) Varying the geofence radius (b) Varying the number of geofences

Fig. 6. The RN and SP metrics for the `LA-MQTT` and *Legacy MQTT* protocols when varying the geofence radius size (R) and the number of LDSs (N) are shown respectively in Figures 6(a) and 6(b).

identification of target MCs for each new data published by a LDS. To this end, we compare the performance of `LA-MQTT` against a legacy MQTT deployment, where the standard MQTT protocol is used without any support for location awareness at both client/backend. In such case, each MC subscribes to one topic among the T available, however no spatial filtering is applied; hence, each MC receives all content published on the topic for which it subscribed, regardless of its actual position and of the geofence region. We denominate this solution as *Legacy MQTT* in the following. Figure 5(a) shows the performance of both `LA-MQTT` and *Legacy MQTT* when varying the number of MCs in the simulation.The y-axis reports the total number of Received MQTT Notifications (shortened to RN in the Figure) from the MCs during the whole simulation. The $y2$-axis reports the SP metric, *i.e.* the ratio of received *relevant* notifications, according to the definition provided in Section 3. Clearly, for both protocols, the RN values increase with the number of MCs subscribing to topics of interest; however, without any filtering mechanism on the backend, the traffic load increases exponentially while the relevance of data – reflected by the SP line – is quite poor. Vice versa, in `LA-MQTT`, only relevant messages are delivered to the MCs; this produces a considerable



(a) (b)

Fig. 7. The *SP* and *SR* metrics as a function of the perturbation index *d* and for three configurations of *R* are shown in Figures 7(a) and 7(b), respectively.

reduction of the network load for the notification process, while the *SP* is near 100%, even considering a relevant number of active users (10000). Figure 5(b) further investigates the scalability of our solution by comparing the average computational load of the LA-MQTT and *Legacy MQTT* protocols when varying the number of MCs. The load is expressed as the average CPU utilization of the MQTT broker. We remark that LA-MQTT does not affect the broker implementation, and hence can be supported by any legacy broker; however, the number of publish-subscribe differs for the LA-MQTT and *Legacy MQTT* protocols. More in details, the LA-MQTT incurs in higher number of spatial notifications, as already shown in Figure 5(a); vice versa, *Legacy MQTT* introduces additional private subscriptions for each MC and topic of interest, as previously illustrated in Section 4.1. Figure 5(b) shows that the resource utilization increases significantly for M=10000, however the performance of LA-MQTT is comparable with *Legacy MQTT*, hence our solution does not introduce significant performance bottleneck compared to the standard. Clearly, on large-scale scenarios, geo-spatial distributed MQTT solutions may be considered to share the load while avoiding the presence of a Single Point of Failure (SPOF); the issue is left as future work and discussed in Section 8. Figure 6(a), which shows the *RN* and *SP* metrics as a function of the geofence radius (R), while M is fixed to 100. For the *MQTT Legacy*, the *RN* metric is not affected by R for the same reason discussed above, whereas in LA-MQTT, the *RN* metric slightly increases with R, since a larger geofence region involves a higher number of potential receivers; however, the *SP* value equals almost 100% for all the configurations of R. Finally, Figure 6(b) shows the *RN* and *SP* metrics for the LA-MQTT and *Legacy MQTT* protocols, when we vary the number of IoT data sources, in our cases constituted by the number of LDSs (N). We set R=300 and M=500. Clearly, the number of geospatial messages to deliver to the MCs grows with N. However, the Figure provides similar insights than the case with varying data receivers (Figure 5(a)), *i.e.*: (i) in absence of spatial filtering mechanisms, the number of notifications grows exponentially for the *Legacy MQTT*; (ii) however, only few of these notifications are spatially relevant for the MC; (iii) vice versa, the *SP* of LA-MQTT is always close to 100% even in dense scenarios with N=2000.

6.3 Impact of Privacy Parameters

In this Section, we evaluate the performance of the privacy mechanisms detailed in Section 5 and, specifically, the impact of parameters d (*i.e.* perturbation index) and b (*i.e.* number of dummy updates) as well as the gain produced

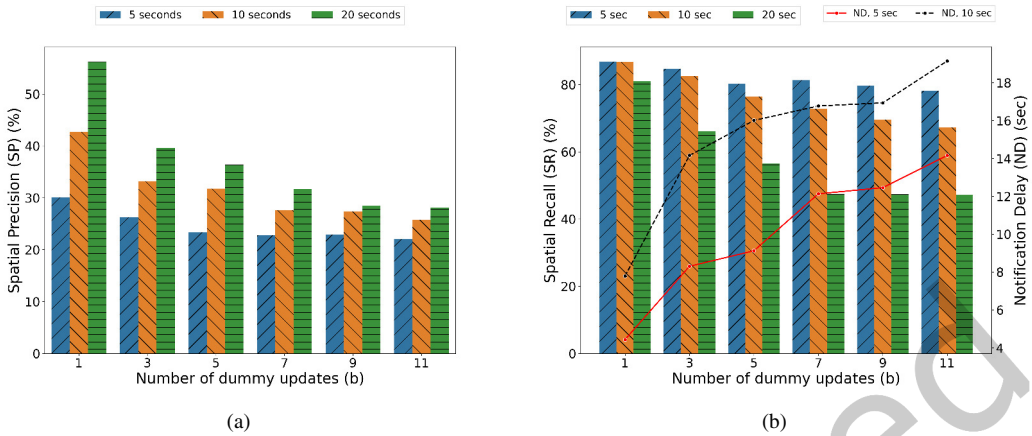
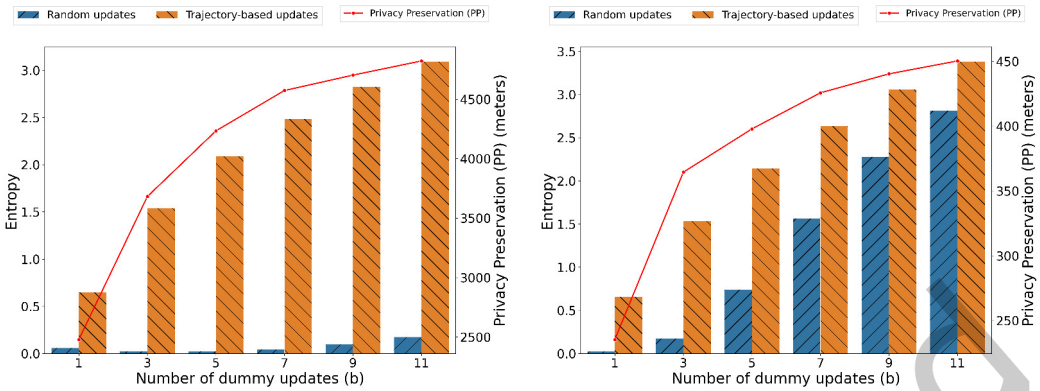


Fig. 8. The SP and SR metrics when varying the number of dummy updates (on the x-axis) and for three configurations of the GPS sampling frequency are shown in Figures 8(a) and 8(b), respectively.

by correlated trajectories (dummy routes). To this end, we consider an incremental evaluation, through which the impact of these three aspects is discussed separately.

Impact of Random Perturbation. We set up a scenario with $M = 50$ and $N = 20$; after sampling its GPS position P_i, each MC_i computes a new position $\tilde{P}_i(d)$ by applying exclusively the *Strategy 1* of the procedure detailed in Section 5. Figure 7(a) shows the impact of different values of the perturbation index d on the SP metric: for each value of d, we consider three configurations of R, respectively equal to 100, 200 and 500 meters. On the same figure, we also depict (on the $y2$-axis) the PP metric, computed as $E[dist(P_i, \tilde{P}_i(d))]$, *i.e.* the average distance between the P_i and $\tilde{P}_i(d)$. It is easy to notice the trade-off between the SP and PP metrics: for $d = 1$, the average PP value almost exceeds the scenario size, while the quality of notifications sent to the MCs is poor. For $d \geq 4$, the random noise has a negligible impact (in the order of few meters), hence we get similar performance than Section 6.2, where privacy mechanisms are disabled. Figure 7(b) depicts the Spatial Recall (SR) over d, again for the three different configurations of R. The SR is computed as the average ratio of cases in which MC_i is within the geofence of a LDS of interest s and receives the corresponding notification, over the total number of cases where $P_i \sqsubset g_s$; hence, differently from the SP metric, it filters out the non-relevantnotifications received by the MC_i. We highlight that both SR and SP metrics reflect the quality of notifications from the MC perspective, although with different meanings: the first metric quantifies the effectiveness of the dissemination process, while the second one takes into account the effort of the dissemination, in terms of network communication overhead and eventual additional geoprocessing required by the MC in order to filter the data. The SR metric follows a similar trend than Figure 7(a), although it increases more sharply for $d \geq 2$, and is more affected by R. The results above further justify the need for a proper tuning of the d parameter.

Impact of Dummy Updates. Next, we evaluate the impact of *Strategy 2*, defined in Section 5. We consider the same scenario of the previous analysis: each MC_i publishes its position every f_{MC} seconds; however, it transmits its real position P_i once every $b + 1$ updates, while in the other b cases it applies the Random Perturbation to P_i, with index equal to d. Figure 8(a) depicts the SP metric when varying the b value on the x-axis, and for three different values of f_{MC} (indicated by the bar color). It is easy to notice that the dummy update mechanism reduces the QoS perceived by the MCs, which might receive spatial notifications not related to their actual position. More



(a) $d = 1$ (b) $d = 2$

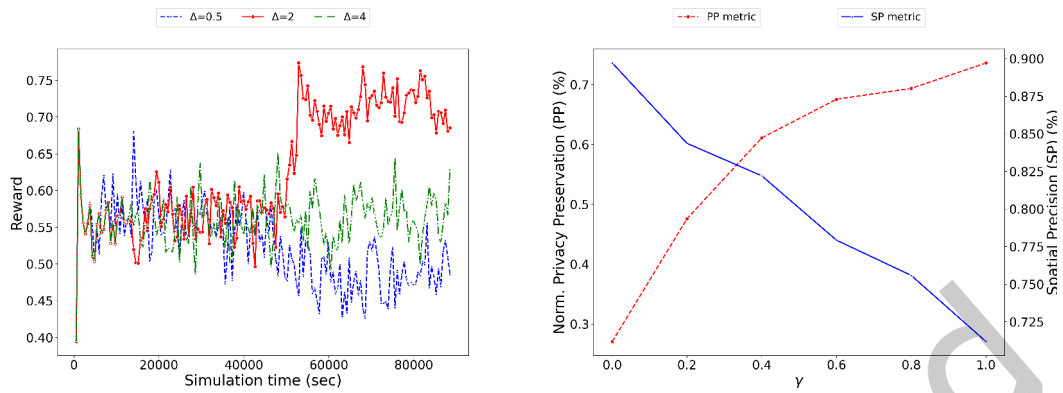
Fig. 9. The entropy and PP metrics as a function of the number of dummy updates in a random Dummy Updates scheme and a trajectory-based one.

in detail, the SP metric decreases as b decreases and/or as f_{MC} decreases. Figure 8(b) shows the impact of b on the SR metric, again for three configurations of f_{MC}. On the $y2$-axis of the same Figure, we depict the Notification Delay (ND), defined as the average time elapsed from the entrance of MC_i in the geofence of a LDS of interest until the reception of the corresponding notification. We can easily notice that: (i) with f_{MC} fixed, the SR decreases as b increases, since a MC might enter and exit a geofence without having the opportunity to publish its true position; (ii) the ND curve increases together with b for all values of f_{MC}, confirming the previous point; (iii) the SR performance worsens when increasing f_{MC}.

Impact of the Dummy Routes. We conclude the analysis by evaluating the usage of correlated trajectories, the ultimate strategy defined in Section 5, on the privacy performance. For the latter, we quantify the risk that the SC identifies the true update from the other b synthetic updates composing a single step. Let $S_i^l = \{\tilde{P^l}_{i,1}, \tilde{P^l}_{i,2}, ..., \tilde{P^l}_{i,b+1}\}$ be the l-th sequence of dummy updates. The privacy risk for MC_i is expressed through the well-known entropy function [36]:

$$H(S_i^l) = \sum_{j=1}^{b+1} PR(\tilde{P^l}_{i,j}) \cdot log_2 \left(PR(\tilde{P^l}_{i,j})\right) \tag{7}$$

where $PR(\tilde{P^l}_{i,j})$ denotes the SC's belief that the true position of MC_i in the j-th value of the sequence. In the optimal case, the dummy values are all equally probable, *i.e.* $PR(\tilde{P^l}_{i,j}) = \frac{1}{b+1}$ $\forall 1 \leq j \leq b$. We compute the $PR(\tilde{P^l}_{i,j})$ value by considering the likelihood of the distance covered between two consecutive sequences, *i.e.* $PR(\tilde{P^l}_{i,j}) = max_{1 \leq k \leq b} PR(dist^*(\tilde{P^l}_{i,j}, \tilde{P^{l-1}}_{i,k}))$, assuming that the distance function $dist^*()$ follows a Gaussian distribution with mean $\phi \cdot \bar{v}$, where ϕ is the interval between two consecutive samples of the same route (at two different steps, therefore $\phi \sim f_i \cdot (b+1)$) and $\bar{v}$ the average speed of the MCs. Figure 9(a) depicts the entropy values when varying the b parameter, and using the maximum perturbation index ($d = 1$); we compare the previous random Dummy Updates scheme, where b updates of the step are generated by applying a random perturbation to the true position (*Strategy 2* of Section 5), against the trajectory-based solution evaluated in this paragraph. On the $y2$-axis we show the average PP metric over b, computed according to Equation 1. It is easy to notice that the entropy of the random solution is low, *i.e.* it is quite easy for the SC to identify the true position of the MC within



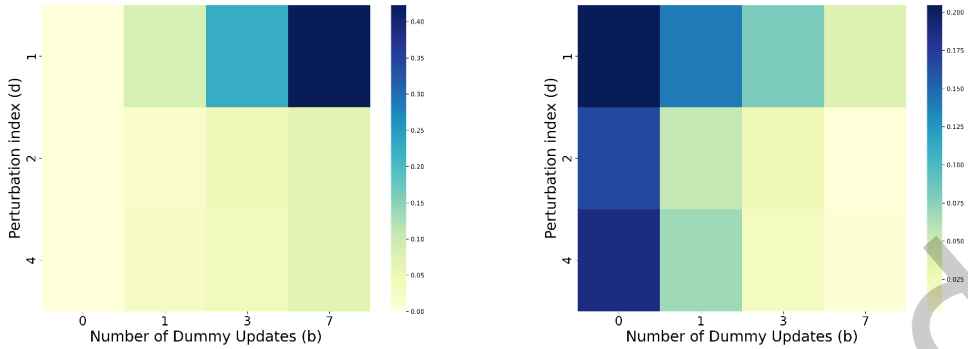
(a) (b)

Fig. 10. The impact of Δ parameter on the exploration-exploitation process is shown in Figure 10(a). The average values of $\bar{PP}$ and SP for different settings of γ parameters are depicted in Figure 10(b).

the sequence of dummy updates generated within a step; this is due to the fact that the average distance among fake updates of two different steps may greatly exceed the expected value $\phi \cdot \bar{v}$. Vice versa, for the trajectory-based solution, the entropy function is maximized by construction, and is not affected by the d parameter. The latter is demonstrated by Figure 9(b) which shows entropy and PP values for $d = 2$. The Figure reveals that entropy performance of the trajectory-based solution does not change significantly, while the performance of the random scheme improves since the random updates are getting closer to the true position; this is a consequence of the reduced amount of perturbation. However, lower values of d determine a reduction of the PP metric, as shown by the red lines in Figures 9(a) and 9(b). The results presented so far demonstrate that the d and b parameters may have a significant impact on SP and PP performance. The optimal tuning of those parameters clearly depend on the desired trade-off, reflected by the γ parameter in Equation 2 but also by scenario-specific conditions such as number and positions of the LDSs. For this reason, in the next section we evaluate the performance of Algorithm 1 enabling the self-tuning of the privacy parameters.

6.4 Performance of the Learning-based Algorithm

In this section we evaluate the performance of the learning-based algorithm described in Section 5.1. Unless stated otherwise, we set the parameters of the simulation environment and of Algorithm 1 as follows: $N = 50$, $M = 50$, R =300m, $T_{max} = 10$, $T_{min} = 0.1$, $\eta = 0.5$, $\alpha = 2$. The simulation time is divided into epochs of length t_e =180s; during an epoch, each MC_i is not allowed to modify its configuration of the privacy parameters (b_i and d_i). In Figure 10(a) we investigate the impact of the parameter controlling the extent of the exploration phase, *i.e.* Δ in Equation 6. To this purpose, the y-axis shows the per-MC value of the instantaneous reward (Equation 2) over the simulation time; the three curves refer to three different configurations of Δ, respectively equal to 0.5, 2 and 4. We set $\gamma = 0.6$. The curves are mostly overlapped till $t = 7500$. When $t > 7500$, the performance of the Algorithm with $\Delta = 0.5$ sharply decreases due to the fact that MCs start exploiting a largely sub-optimal configuration, a clue that the exploration phase has been too short. In opposite way, for $\Delta = 4$, the reward function does not show significant changes over time since the exploration phase exceeds the simulation length, and hence the system never enters the exploitation phase. For $\Delta = 2$, the learning process is effective since the average reward increases at $t > 10000$, *i.e.* when the system starts reinforcing the optimal solution found so far; for this reason, we use $\Delta = 2$ for the further



(a) Maximizing the PP (γ=1)

(b) Maximizing the SP (γ=0)

Fig. 11. The average probabilities of privacy configuration selection (b on the row, d on the column) during the exploitation phase are depicted for γ=1 (on the left) and γ=0 (on the right).

analysis. Figure 10(b) depicts the two components of the reward function of Equation 2, *i.e.* the normalized PP ($\bar{PP}$) on the y-axis and the SP on the $y2$-axis, when varying the γ coefficient[4]; we recall that γ must be tuned by the end-users and controls the desired trade-off between the quality of the notifications and location privacy. For $\gamma = 1$, the system goal is to maximize the $\bar{PP}$ only, regardless of the SP; the opposite occurs for $\gamma = 0$. We can notice that the two lines follow the expected behavior hence we can conclude that Algorithm 1 is able to capture the PP-SP trade-off in an effective manner. In Figures 11(a) and 11(b) we provide further evidence of the ability of Algorithm 1 to adapt the client configuration according to the MC requirement, *i.e.* the γ value. The two heatmaps show the experimental probability (*i.e.* the usage frequency) for each configuration of the privacy parameters, computed as proportion of epochs, again during the exploitation phase, in which each configuration has been adopted by the MCs. For the dummy updates, we considered 4 configurations of parameter b on the x-axis, *i.e.* 0 (no dummy), 1, 3, 7. Similarly, we considered 3 configurations of parameter d on the y-axis, *i.e.* 1, 2, 4. The heatmap of Figure 11(a) refers to $\gamma = 1$, *i.e.* privacy maximization. In accordance with the analysis presented in Section 6.3, we can notice that the MCs mostly employ configurations with maximum number of dummy updates ($b \geq 4$) and maximal perturbation of the GPS position ($d = 1$). On the contrary, the heatmap of Figure 11(b) refers to $\gamma = 0$, *i.e.* quality of notification maximization. In this case, the MCs mostly rely on configurations with dummy update mechanism disabled ($b = 0$, while d is not effective in this case). Although the distributions are not uniform over the available configurations, we can notice that sub-optimal configurations may be still selected with probability greater than zero, for both $\gamma = 0$ and $\gamma = 1$; this is a joint effect of the probabilistic selection mechanism and of the exploration phase, which is progressively discounted over time but not below a minimal threshold $T_{min} > 0$, in order to ensure system adaptivity to environmental changes.

7 USE-CASE

In order to test the functionalities and the efficiency of our proposal, we implemented a Proof-of-Concept (PoC) prototype that features all the components presented throughout the paper. More in detail, the architecture of the PoC is presented in Figure 12, where the main entities are highlighted. In particular, we deployed an instance of the

[4]We considered the average metric values during the exploitation phase only, *i.e.* t >50000.

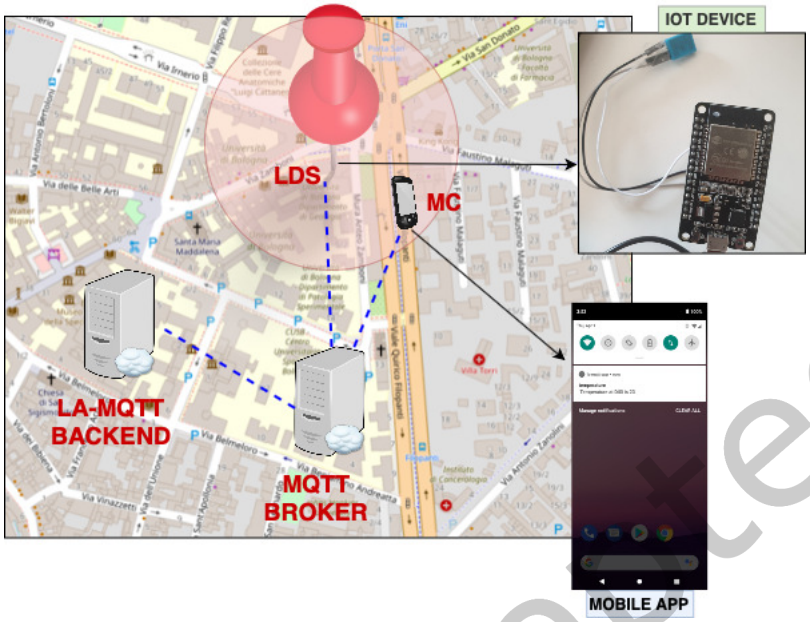


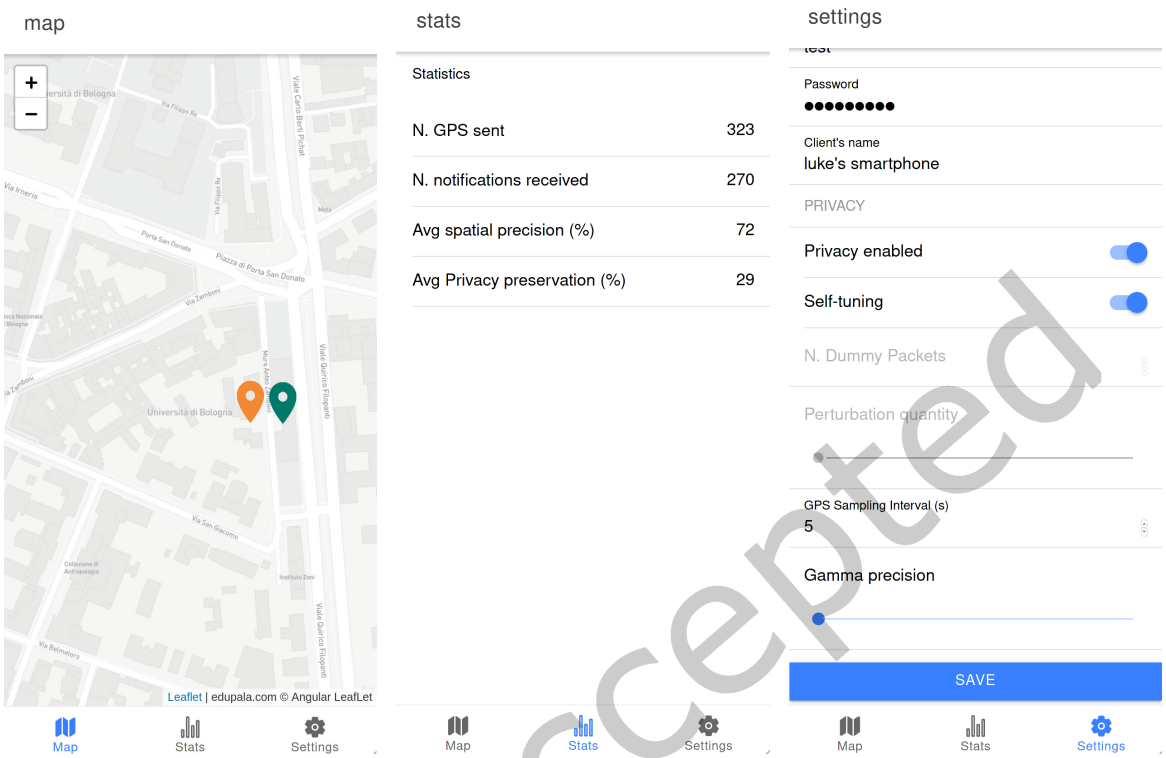
Fig. 12. Architecture of our PoC implementation of a LA-MQTT scenario with a MC and a LBS.

Mosquitto MQTT broker into a server as well as our implementation of the LA-MQTT backend. Once the backend attaches to Mosquitto, these two server instances are considered a single entity, as made clear in the previous sections. Furthermore, we used our implementation of the LA-MQTT client library to develop two entities: a LBS and a MC. Both the library and the backend are the same as the one used in Section 6 and are a final release version, therefore they should not be considered a PoC, neither a prototype.

In our PoC scenario, the LBS is represented by an IoT device that periodically published temperature values. The picture of the prototype can be found within Figure 12. In particular, it is composed by a microcontroller, in this case we used an Espressif ESP32 board[5] with an ESP-12 WiFi module, and a DHT-11 temperature and humidity sensor. The IoT Module is flashed with a C sketch via the Arduino IDE, is deployed at the Department of Computer Science and Engineering of the University of Bologna, and it has access to the internet via its WiFi module. The IoT device uses the LA-MQTT client library to perform the "Geofence publish" outlined in Section 4 instead of the normal MQTT publish, and performs such operation with a frequency f_{temp} =5s. On the other hand, the MC is represented by a Mobile app for Temperature monitoring that interacts with the MQTT Broker via the LA-MQTT Client Library. The Mobile app is a hybrid Ionic Angular app[6] written in Typescript and built natively via Capacitor[7]. In particular, as instructed in Section 4, it periodically publishes its position (which may be blurred via our privacy preserving techniques) and performs the "Topic subscription" operation, where t_s is our use-case-defined temperature topic. Ideally, upon entering the geofence of the LBS, the application starts receiving temperature updates and notifies the user via a system notification, as its screenshot within Figure 12 suggests.

An overview of the mobile application is shown in Figure 13 via screenshots. In particular, the app consists of three main screens, accessible through the bottom navigation bar. The first screen, shown in Figure 13(a), displays

[5] https://www.espressif.com/en/products/socs/esp32
[6] https://ionicframework.com/
[7] https://capacitorjs.com/



(a) Map (b) Statistics (c) Settings

Fig. 13. The three main screens of the mobile application that acts as a MCs in our PoC scenario.

the position of the user itself (green marker) and any other LBS (orange marker, in our case only one is represented). In our case, the LBS has a circular geofence with radius 300m, which means that the MC is close enough to get notifications. The third screen, shown in Figure 13(c), displays the settings and all the parameters that the user of the app can change manually. Specifically, the application envisions a login, where MCs uniquely identify themselves against the MQTT broker. With respect to the privacy mechanisms illustrated in Section 5, the final user can choose to activate the privacy preservation through a switch. If privacy is enabled, then the user may choose to let the system tune the parameters through reinforcement learning, as in Section 5.1, by toggling on the "Self-tuning" switch. Conversely, if the switch is off, then the user can manually define the spatial perturbation and the number of dummy routes (disabled otherwise). Other configurable parameters include the position publish interval f_{MC} and the trade-off parameter γ presented in Equation 2. The second screen shows the statistics in terms of LA-MQTT messages sent and received as well as the estimated SP and PP metrics. In particular, the example in Figure 13(b) shows the number of position publish sent and the number of notifications received. In this case, the privacy preservation relies on top of the reinforcement learning algorithm, which leads the Spatial Precision to be 72% and the Privacy Preservation to be 29%.

Furthermore, we conducted an experiment in order to evaluate the real delay occurring in our PoC scenario. In detail, the mobile app detects locally when entering the geofence of the LBS and, from then, it measures the time

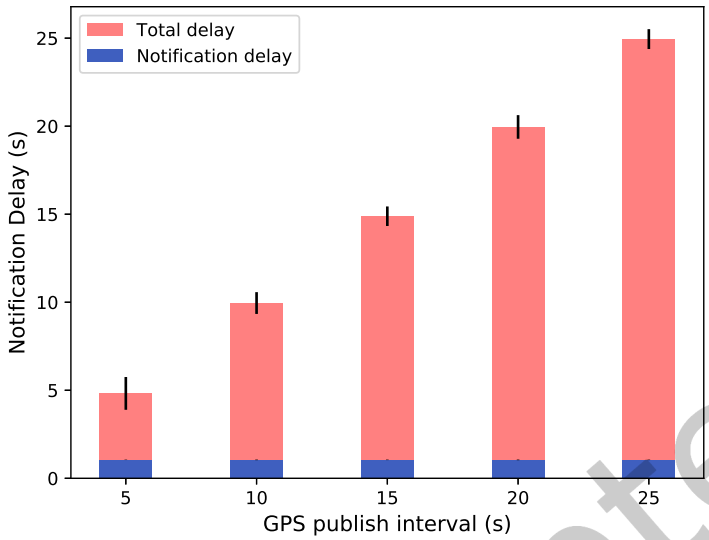


Fig. 14. Delay measured by the mobile app for different values of f_{MC} from the time the user enters the geofence until the notification is received.

taken to receive the relative notification. Figure 14 shows the results of the tests we have conducted, in which each bar corresponds to such delay. We set five different values of the position publish interval f_{MC}. More specifically, we split the delay in the figure into two phases: the blue bottom bar is the "notification delay", *i.e.* the delay occurring from the position publish operation till the reception of the notification, while the red top bar is the difference between the total delay and the notification delay. The results are averaged over multiple measurements for each value of f_{MC}. It is easy to see that the notification delay is unaffected by f_{MC}, as it is entirely due to the network connection quality. On the other hand, the total delay increases linearly with f_{MC}, which is the dominating factor.

8 CONCLUSIONS

In this paper, we have proposed `LA-MQTT`, an extension of MQTT to support location-aware communications for IoT scenarios. The extension has been demonstrated to be effective and easily pluggable into a preexisting MQTT-driven scenario, being broker-agnostic and not requiring any modification to the standard protocol. Through `LA-MQTT` we showed how Publishers can specify an area of influence and how Subscribers can be notified only when certain conditions against such area are met (*e.g.* being physically inside it). We have shown how this implies that the `LA-MQTT` Subscribers publish periodically their position, which could result in a privacy concern. Consequentially, we proposed a simple yet effective privacy-preserving paradigm that is based on dummy routes and multiple basic privacy self-assessment schemes. The trade-off between privacy preservation and spatial precision metrics has been addressed through a novel reinforcement learning algorithm. Finally, we have shown the efficiency and soundness of the proposal via extensive hybrid simulations feeding the `LA-MQTT` architecture with synthetic data. Furthermore, we implemented a real Proof-of-Concept prototype to showcase the feasibility of our proposal in a real-world IoT scenario.

8.1 Future Works

The work presented in this paper has the potential of being expanded in several directions. We list the main current and future activities below.

Evaluation via IoT datasets. The current evaluation methodology of LA-MQTT is based on hybrid simulation, with the front-end and backend components being fed with synthetic data produced by a mobility simulator. At the same time, many open datasets are publicly available for the IoT domain, including mobility traces and location-based environmental datasets. As a future work, we aim to validate LA-MQTT operations by considering a restricted number of IoT datasets to setup the scenario and to model the behaviours of the MCs and LDSs. The choice might not affect the performance of LA-MQTT but it may contribute to increase the realism of the evaluation.

Beyond MQTT. LA-MQTT is highly based on legacy MQTT in order to maximize its ease of use and to speed up its adoption by a legacy environment. In fact, LA-MQTT can be adopted without redesigning a preexisting scenario from scratch, rather, it can be on-boarded at runtime. Nonetheless, we observe that the LA-MQTT workflow can be adopted by any technology with publish-subscribe primitives. In the short run, we aim to evaluate its adaptability by taking into account a generic definition of the pub/sub paradigm.

Enhance the Privacy Protection. In order to add the privacy preservation into the LA-MQTT framework, we presented in this paper two privacy-aware mechanisms that have been consistently used in the past. However, the algorithms themselves are not the primary focus of the paper, rather, they display the inclusion of privacy as a main pillar of LA-MQTT. In fact, as a future work, we aim to validate LA-MQTT by taking into account a grater pool of privacy-aware mechanisms, as well as enhancing further the ones that we presented here. For instance, Dummy Updates could be perfected by adding background information in the route generation phase in order to make dummies more realistic.

Support for distributed architectures of brokers. The centralized MQTT architecture may pose scalability issues in massive IoT scenarios where all devices connect to a single broker. For this reason, distributed multi-broker solutions are under investigations to increase system reliability and availability. In addition, proper load balancing policies are needed to avoid performance bottlenecks and to minimize inter-broker traffic overhead [14]. The LA-MQTT solution may benefit of such distributed architectures in order to support IoT applications related to smart-cities, such as the environmental monitoring use-case described in Section 7. Moreover, implicit characteristics of the LA-MQTT scenario, such as the knowledge of the positions of clients and the locality of geofencing communications, can be exploited to design spatial-aware balancing techniques among the brokers.

REFERENCES

[1] Enas Ahmad, Maha Alaslani, Fahad R. Dogar, and Basem Shihada. 2020. Location-Aware, Context-Driven QoS for IoT Applications. *IEEE Systems Journal* 14, 1 (2020), 232–243. https://doi.org/10.1109/JSYST.2019.2893913

[2] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys Tutorials* 17, 4 (2015), 2347–2376. https://doi.org/10.1109/COMST.2015.2444095

[3] Stefano Alletto, Rita Cucchiara, Giuseppe Del Fiore, Luca Mainetti, Vincenzo Mighali, Luigi Patrono, and Giuseppe Serra. 2016. An Indoor Location-Aware System for an IoT-Based Smart Museum. *IEEE Internet of Things Journal* 3, 2 (2016), 244–253. https://doi.org/10.1109/JIOT.2015.2506258

[4] Mohammad Abu Alsheikh, Yutao Jiao, Dusit Niyato, Ping Wang, Derek Leong, and Zhu Han. 2017. The accuracy-privacy trade-off of mobile crowdsensing. *IEEE Communications Magazine* 55, 6 (2017), 132–139.

[5] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. 901–914.

[6] Aleksandar Antonić, Martina Marjanović, Krešimir Pripužić, and Ivana Podnar Žarko. 2016. A mobile crowd sensing ecosystem enabled by CUPUS: Cloud-based publish/subscribe middleware for the Internet of Things. *Future Generation Computer Systems* 56 (2016), 607–622.

[7] Louise Barkhuus and Anind K Dey. 2003. Location-Based Services for Mobile Telephony: a Study of Users' Privacy Concerns.. In *Interact*, Vol. 3. Citeseer, 702–712.

[8] Robert Bryce, Thomas Shaw, and Gautam Srivastava. 2018. Mqtt-g: A publish/subscribe protocol with geolocation. In *2018 41st international conference on telecommunications and signal processing (TSP)*. IEEE, 1–4.

[9] Bertil Chapuis, Benoît Garbinato, and Lucas Mourot. 2017. A horizontally scalable and reliable architecture for location-based publish-subscribe. In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 74–83.

[10] Yanzhe Che, Qiang Yang, and Xiaoyan Hong. 2012. A dual-active spatial cloaking algorithm for location privacy preserving in mobile peer-to-peer networks. In *2012 IEEE Wireless Communications and Networking Conference (WCNC)*. 2098–2102. `https://doi.org/10.1109/WCNC.2012.6214137`

[11] Lisi Chen, Shuo Shang, Zhiwei Zhang, Xin Cao, Christian S Jensen, and Panos Kalnis. 2018. Location-aware top-k term publish/subscribe. In *2018 IEEE 34th international conference on data engineering (ICDE)*. IEEE, 749–760.

[12] Chi-Yin Chow, Mohamed F Mokbel, and Xuan Liu. 2011. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. *GeoInformatica* 15, 2 (2011), 351–380.

[13] Cisco. 2018. *Annual Internet Report (2018–2023) White Paper*. Technical Report.

[14] Andrea Detti, Ludovico Funari, and Nicola Blefari-Melazzi. 2020. Sub-Linear Scalability of MQTT Clusters in Topic-Based Publish-Subscribe Applications. *IEEE Transactions on Network and Service Management* 17, 3 (2020), 1954–1968. `https://doi.org/10.1109/TNSM.2020.3003535`

[15] Jasenka Dizdarević, Francisco Carpio, Admela Jukan, and Xavi Masip-Bruin. 2019. A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. *ACM Computing Surveys (CSUR)* 51, 6 (2019), 1–29.

[16] Andrew Fisher, Gautam Srivastava, and Robert Bryce. 2019. MQTTg: An Android Implementation. In *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*. 140–144. `https://doi.org/10.1109/TSP.2019.8769102`

[17] Fernando Fontes, Bruno Rocha, Alexandre Mota, Paulo Pedreiras, and Valter Silva. 2020. Extending MQTT-SN with Real-Time Communication Services. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vol. 1. 1–4. `https://doi.org/10.1109/ETFA46521.2020.9212147`

[18] Aris Gkoulalas-Divanis, Panos Kalnis, and Vassilios S Verykios. 2010. Providing k-anonymity in location based services. *ACM SIGKDD explorations newsletter* 12, 1 (2010), 3–10.

[19] Long Guo, Lu Chen, Dongxiang Zhang, Guoliang Li, Kian-Lee Tan, and Zhifeng Bao. 2015. Elaps: An efficient location-aware pub/sub system. In *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 1504–1507.

[20] Jonathan Hasenburg and David Bermbach. 2020. GeoBroker: Leveraging geo-contexts for IoT data distribution. *Computer Communications* 151 (2020), 473–484.

[21] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. 2008. MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks. In *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*. IEEE, 791–798.

[22] Felicien Ihirwe, Giovanni Iovino, and Davide Di Ruscio. 2021. Towards an MQTT5 geo-location extension for location-aware applications. In *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*. IEEE, 100–105.

[23] ISO/IEC 20922:2016 2016. *Information technology — Message Queuing Telemetry Transport (MQTT) v3.1.1*. ISO/IEC Standard. International Organization for Standardization.

[24] Ryo Kawaguchi and Masaki Bandai. 2019. A distributed MQTT broker system for location-based IoT applications. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 1–4.

[25] Ryo Kawaguchi and Masaki Bandai. 2020. Edge based MQTT broker architecture for geographical IoT applications. In *2020 International Conference on Information Networking (ICOIN)*. IEEE, 232–235.

[26] Hidetoshi Kido, Yutaka Yanagisawa, and Tetsuji Satoh. 2005. An anonymous communication technique using dummies for location-based services. In *ICPS'05. Proceedings. International Conference on Pervasive Services, 2005.* IEEE, 88–97.

[27] Anthony LaMarca and Eyal De Lara. 2008. Location systems: An introduction to the technology behind location awareness. *Synthesis Lectures on Mobile and Pervasive Computing* 3, 1 (2008), 1–122.

[28] Hai Liu, Xinghua Li, Hui Li, Jianfeng Ma, and Xindi Ma. 2017. Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 1–9.

[29] Jorge E. Luzuriaga, Miguel Perez, Pablo Boronat, Juan Carlos Cano, Carlos Calafate, and Pietro Manzoni. 2015. A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks. In *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*. 931–936. `https://doi.org/10.1109/CCNC.2015.7158101`

[30] Lukas Malina, Gautam Srivastava, Petr Dzurenda, Jan Hajny, and Radek Fujdiak. 2019. A secure publish/subscribe protocol for internet of things. In *Proceedings of the 14th international conference on availability, reliability and security*. 1–10.

[31] Stefan Mijovic, Erion Shehu, and Chiara Buratti. 2016. Comparing application layer protocols for the Internet of Things via experimentation. In *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. 1–5. https://doi.org/10.1109/RTSI.2016.7740559

[32] Biswajeeban Mishra and Attila Kertesz. 2020. The Use of MQTT in M2M and IoT Systems: A Survey. *IEEE Access* 8 (2020), 201071–201086.

[33] Steven Donald Monday and Joshua Robert Dalcher. 2010. Geofence system with integrated user interface. US Patent App. 12/222,710.

[34] Federico Montori and Luca Bedogni. 2020. A Privacy Preserving Framework for Rewarding Users in Opportunistic Mobile Crowdsensing. In *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 1–6.

[35] Federico Montori, Luca Bedogni, and Luciano Bononi. 2017. A collaborative internet of things architecture for smart cities and environmental monitoring. *IEEE Internet of Things Journal* 5, 2 (2017), 592–605.

[36] Ben Niu, Sheng Gao, Fenghua Li, Hui Li, and Zongqing Lu. 2016. Protection of location privacy in continuous LBSs against adversaries with background information. In *2016 International conference on computing, networking and communications (ICNC)*. IEEE, 1–6.

[37] Ben Niu, Qinghua Li, Xiaoyan Zhu, Guohong Cao, and Hui Li. 2015. Enhancing privacy through caching in location-based services. In *2015 IEEE conference on computer communications (INFOCOM)*. IEEE, 1017–1025.

[38] Xiao Pan, Jianliang Xu, and Xiaofeng Meng. 2012. Protecting Location Privacy against Location-Dependent Attacks in Mobile Services. *IEEE Transactions on Knowledge and Data Engineering* 24, 8 (2012), 1506–1519. https://doi.org/10.1109/TKDE.2011.105

[39] Charith Perera, Dumidu S. Talagala, Chi Harold Liu, and Julio C. Estrella. 2015. Energy-Efficient Location and Activity-Aware On-Demand Mobile Distributed Sensing Platform for Sensing as a Service in IoT Clouds. *IEEE Transactions on Computational Social Systems* 2, 4 (2015), 171–181. https://doi.org/10.1109/TCSS.2016.2515844

[40] Aniket Pingley, Wei Yu, Nan Zhang, Xinwen Fu, and Wei Zhao. 2009. CAP: A context-aware privacy protection system for location-based services. In *2009 29th IEEE International Conference on Distributed Computing Systems*. IEEE, 49–57.

[41] Eun-Jeong Shin, Dhrubajyoti Ghosh, Apoorva Gupta, Gurinder Pal Singh, Navneet Joshi, and Alfred Kobsa. 2016. Message of interest: A framework of location-aware messaging for an indoor environment. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. 1–4. https://doi.org/10.1109/PERCOMW.2016.7457065

[42] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. 2011. Quantifying location privacy. In *2011 IEEE symposium on security and privacy*. IEEE, 247–262.

[43] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA.

[44] Sasu Tarkoma. 2012. *Publish / Subscribe Systems: Design and Principles* (1st ed.). Wiley Publishing.

[45] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Keng-Yan Tan. 2014. Performance evaluation of MQTT and CoAP via a common middleware. In *2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP)*. IEEE, 1–6.

[46] Muhammad Usman, Muhammad Rizwan Asghar, Imran Shafique Ansari, Fabrizio Granelli, and Khalid A Qaraqe. 2018. Technologies and solutions for location-based services in smart cities: Past, present, and future. *IEEE Access* 6 (2018), 22240–22248.

[47] Riccardo Venanzi, Burak Kantarci, Luca Foschini, and Paolo Bellavista. 2018. MQTT-driven node discovery for integrated IoT-fog settings revisited: The impact of advertiser dynamicity. In *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*. IEEE, 31–39.

[48] Christopher J. C. H. Watkins and Peter Dayan. [n.d.]. Q-learning. *Machine Learning* 3 ([n. d.]), 279–292. https://doi.org/10.1007/BF00992698

[49] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. 2014. Internet of things for smart cities. *IEEE Internet of Things journal* 1, 1 (2014), 22–32.

[50] Ying Zhong, Shunzhi Zhu, Yan Wang, Jianmin Li, Xinxin Zhang, and Jedi S Shang. 2020. Pairwise Location-Aware Publish/Subscribe for Geo-Textual Data Streams. *IEEE Access* 8 (2020), 211704–211713.