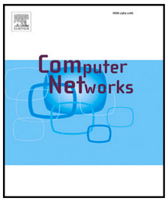
Computer Networks 243 (2024) 110301

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



A hierarchical distributed trusted location service achieving location k-anonymity against the global observer

Francesco Buccafurri [a,*], Vincenzo De Angelis [b], Maria Francesca Idone [a], Cecilia Labrini [a]

[a] DIIES Dept., University Mediterranea of Reggio Calabria, Via dell'Universita 25, Reggio Calabria, 89124, Italy
[b] DIMES Dept., University of Calabria, Via Pietro Bucci, Rende, 87036, Italy

ARTICLE INFO

Keywords:
Location *k*-anonymity
Privacy
Location-based services

ABSTRACT

As widely known in the literature, location-based services can seriously threaten users' privacy. Privacy-aware location-based services can be obtained by protecting the user's identity, so that queries cannot be linked with users. A way to do this is to place a trusted third party, called *Location Trusted Service*, between the user and the service provider, with the role of mediating the queries coming from the users and proxying them to the provider. Before proxying the query, the Location Trusted Service builds a *cloaking area* that includes a sufficient number of users such that it can represent an anonymity set. This way, the identity of the user is protected against an untrusted service provider. Unfortunately, in wide-area scenarios, a centralized location-trusted service might represent a serious threat to security and privacy because the service represents a single point of failure that manages very critical and massive information. Moreover, privacy protection also against a global adversary capable to monitor the whole traffic, would result in an excessive amount of cover traffic in the network (being cover traffic necessary in this threat model).

To overcome the above limitations we propose a hierarchical Location Trusted Service, whose implementation benefits from the edge–cloud paradigm. In our proposal, the territory is organized in hierarchical *zones* possibly managed by different autonomous organizations. Organizations that manage higher zones are involved when lower-level organizations are not able to satisfy the requests of the users. As only the lowest-level services manage exact location data, while the higher ones operate only on aggregate values, the risk associated with the single point of failure of the centralized solution is drastically reduced. Moreover, leveraging the edge–cloud implementation of the system, network traffic is better confined to the edge of the network, making the protection against the global observer feasible.

A nice feature of our method is that it is parametric with respect to any existing cloaking area construction technique. However, as our method, for non-local queries, operates on aggregate data, a certain degree of approximation is introduced. To validate our proposal, we conducted an experimental campaign on a real-life map by applying our method on top of well-known cloaking area construction technique called Casper. The results turned out to be positive. For a wide range of sizes of the anonymity set, the approximation (expressed by the metric called effectiveness) is less than 10%. On the other hand, concerning the network performance, we have observed an improvement in latency and throughput ranging from 20% to 170% (depending on the size of the anonymity set). In the highest density distribution, we achieve a 66% saving in overall (non-local) traffic compared to the centralized approach.

1. Introduction

Location-based services (LBS) occupy an important position within pervasive, ubiquitous, and wide-area computing systems. These services encompass various types, including navigational services, resource discovery (typically points of interest), traffic updates, news, weather, emergency alerts, advertising, location-based games, and so on [1–4]. They can be continuous (such as navigational services), may require different localization precision, and can be delivered as push services. However, LBS may represent a serious threat to people's privacy [5]. Indeed, the link between the content of the required service and the user's location may enable an honest-but-curious provider to associate the user's identity with sensitive information such as habits, health status, religion, or sexual orientation. This is a very well-known problem in the literature [6–9], and basically it depends on the fact that

[*] Corresponding author.
E-mail address: bucca@unirc.it (F. Buccafurri).

https://doi.org/10.1016/j.comnet.2024.110301
Received 21 February 2024; Received in revised form 25 February 2024; Accepted 29 February 2024
Available online 1 March 2024

location data are *quasi-identifiers*, allowing the adversary to discover the identity of the victim, if combined with background knowledge or through collusion among different adversary parties.

One of the approaches used to contrast the above problem is to protect the user's identity. The goal is to prevent location-based queries from being linked to user identities. This can be obtained by relying on a Trusted Third Party, named *Location Trusted Service* (LTS, for short). The LTS acts as an intermediary for queries originating from users, forwarding them to the LBS provider. However, each query is not forwarded as it is. Instead of the exact user position, the LTS builds a *cloaking area* including at least k users and such that, for each of these k users, the associated cloaking area is the same. This way, *location k-anonymity* [6–8] is achieved. The stability condition of the cloaking area, called *reciprocity* [10,11] is fundamental, because it prevents reverse-engineering attacks [11] that can reduce actual anonymity. Cloaking areas should also satisfy the *effectiveness* property [10] that requires minimizing their extension. Indeed, large cloaking areas incur high processing overhead from the side of LBS and network costs, due to the high number of candidate results to return to the LTS.

In this paper, we face the problem of implementing the LTS approach in a wide-area scenario, which is recognized as a non trivial task [12].

The basic research question of this paper is the following. How can we avoid a centralized LTS provider that traces the movements of all the people and represents a single point of failure respect security and privacy threats?

The immediate answer to the above question is to distribute the service among possible different organizations with a small territorial competence.

However, the implementation of such distribution is not trivial. Indeed, a cloaking area could overlap different competence zones. Therefore, a hierarchical organization would be necessary, in such a way that, when overlapping occurs, the task is forwarded to the higher-level LTSs. Unfortunately, this would vanish the benefits of splitting the competence of LTSs. Indeed, the root of the hierarchy would have the same power as the centralized LTS.

The basic idea of this paper is then to give the power of detailed tracing only to LTSs of the lowest level, which have competence on very small areas. The higher LTSs can access only aggregate location data, while detailed user position data are exclusively managed by local LTSs. This drastically reduces the risk (because drastically reduces the impact) of a security incident in which one LTS is compromised, with respect to the centralized approach in which one LTS manages huge volumes of location data referring to a large population and a wide geographical area.

But, to do this, we have to solve another non-trivial problem. Given any state-of-the-art cloaking-area construction algorithm, able to return a *good* cloaking area (thus fulfilling the properties called (called *reciprocity* and *effectiveness*) built on the basis of the query of a user, how it can be used on aggregate location data preserving its properties and thus guaranteeing valid cloaking areas also on aggregate data?

Moreover, our solution addresses also another important issue. Indeed, any LTS-based approach becomes ineffective when faced with a global passive adversary [13] capable of monitoring the flow of messages. In such a scenario, the source of the query can be identified among the anonymity set of users. In real-life contexts, this is for example the case in which we want to provide LBS completely within a mobile social network. In this case, the LTS could be an entity that interacts with the users and the LBS provider (possibly, the social network itself) by using communication mechanisms provided by the social network (and thus completely observable by the provider).

In our paper, thanks to the hierarchical organization, we are also able to contrast the global passive adversary in a much more feasible way than with a centralized organization. It is well-known in the literature that resistance to the global passive adversary requires the inclusion of cover traffic [14,15]. However, the introduction of cover traffic in the entire network (needed in the centralized approach) is obviously not feasible. The hierarchical organization enables us to design a solution in which cover traffic is introduced only in the network segment closest to the user, leveraging an edge–cloud implementation of the hierarchical system [16]. In other words, if local LTSs are placed at the edge of the network, cover traffic is limited at this level, drastically reducing the traffic overhead with respect to the centralized approach. Specifically, we use position notification as cover traffic to hide queries and multicast to hide responses against the global adversary. Placing local LTSs on the edge of the network, also gives benefits in terms of the actual traffic, because detailed data (much more voluminous than the aggregate data) are confined only to the edge of the network. In the rest of the network, only aggregate data are transmitted. We also performed an experimental campaign to show that the overall approach is feasible from the side of network performance.

As a final contribution, we show that the innovative LTS system can enable new multi-organization business models (possibly operating selectively on some services), thus opening new information-technology markets.

Once we framed above our proposal within a sketch of the background knowledge (LBS), the definition of the problem we want to deal with (privacy protection also against the global passive adversary), the disadvantages of the existing methods (infeasibility of centralized LTS and poor effectiveness against the global observer), and a description of our proposal, that highlights how we overcome such disadvantages, we make the following important considerations.

Our paper falls within the field of location k-anonymity, aiming to preserve privacy in Location-Based Services (LBS). It is noteworthy that, despite the emerging research direction emphasizing the use of differential privacy for privacy in location-based services [17], location k-anonymity remains a valid approach when the goal is to protect the user's identity [7,18–21].

Another important aspect to observe is that when dealing with k-anonymity-based approaches (in a more general setting than location privacy-preserving techniques), some drawbacks have to be considered [22]. However, k-anonymity is still alive and effective as evidenced in some recent works [23–26]. Moreover, location k-anonymity deserves specific considerations, as recently claimed in [27]. Indeed, in [27], when referring to location k-anonymity, the authors say that even though other techniques, such as l-diversity, t-closeness [28], and so on strive to protect query privacy, they are strictly related to query content and are outside the scope of location privacy-preserving techniques.

The structure of the paper is the following. Some basic notions useful for the comprehension of the paper are given in Section 2. In Sections Section 3, we describe the literature related to our work. Section 4 introduces the mechanism we propose to construct an approximate cloaking area. In Section 5, we describe our LTS distributed and hierarchical system. In Section 6, an enhanced more practical model is provided, introducing multiple services and competence overlap among LTSs, and discussing the implications of this enhancement. This section also includes a discussion about some implementation aspects and a computational complexity evaluation. A possible business model associated with our system is proposed in Section 7. The security of the solution is analyzed in Section 8. The experimental validation of the proposed approach is provided in Section 9. Finally, in Section 10, we draw our conclusions.

2. Background

In this section, we provide the background knowledge useful for understanding the remainder of the paper. Specifically, we give the fundamentals of the location-trusted-service-based approach used to safeguard privacy in location-based services.

Location-based services (LBS) are services that rely on the location of users, and they can pose serious privacy threats [5]. Various approaches aim to contrast this problem. This paper refers to the approach aimed at protecting the user's identity [26]. This approach is based on the presence of a *Trusted Third Party* (TTP), called *Location Trusted Service* (LTS), which plays the role of anonymizer of the user's requests towards the LBS provider [29]. The LTS receives the query from the user and, instead of the exact position, sends a cloaking area to the LBS provider including at least k users (including the requester), in such a way that *location k-anonymity* [8] is achieved. This means that the probability for the adversary (i.e. the LBS provider) to identify the user requesting a location-based service is at most $\frac{1}{k}$, provided that an additional feature (detailed below) is adopted.

To illustrate the approach based on the location k-anonymity, let us consider the following example. Suppose that Bob wants to know the points of interest close to his position by relying on an LBS provider. To prevent an honest-but-curious LBS provider from learning information about Bob related to the query, a possible way is to protect Bob's identity, so that the LBS query cannot be linked to him. Unfortunately, the use of anonymous IDs is not enough. Indeed, the location itself is a *quasi-identifier* [6] and, therefore, allows the LBS provider to identify Bob if the location data are combined with other public data, background knowledge, or through collusion with external parties. To avoid this, the LTS is placed in the middle, between Bob and the LBS provider. The query is submitted by Bob to the LTS (playing as a Trusted Third Party), along with his exact position. The LTS builds a cloaking area, including an anonymity set of users (AS), which Bob belongs to. If the required privacy level is k, then the cardinality of the anonymity set must be no smaller than k. These users are selected so that, based on the knowledge available to the LBS provider, they are indistinguishable for the provider. The LTS removes the user ID, and submits the query to the LBS provider along with the cloaking area, instead of the exact position. This is done with the objective of obtaining the probability for the adversary (i.e., the LBS provider) to identify Bob and link him to the query is not higher than $\frac{1}{k}$, which is acceptable for a sufficiently large value of k. The LBS sends the answer to Bob's query to the LTS, which can filter it based on Bob's exact location, and sends the refined response to Bob to minimize client-side communication overhead.

The above intuitive description leads to two important formal definitions (see below). The first is the notion of *cloaking area*. The second is the notion of *reciprocity*. A cloaking area, constructed by the LTS, is dependent on the user's required level of privacy. We recall that, the privacy level refers to the denominator k of the fraction $1/k$ which measures the probability of an attacker to de-anonymize the query. Specifically, the cloaking area is a region A that includes at least k users. The objective is to transmit to the LBS provider no information allowing it to distinguish these users from each other, thus achieving the required privacy level. However, to be effective, reciprocity is necessary. Reciprocity implies that, for any user u' involved in the cloaking area built around the inquiring user u, the cloaking area built around u' must include the same users as the cloaking area built for u. Indeed, it is easy to see that if reciprocity does not hold, intersection attacks are possible. These attacks enable the attacker to reduce the actual privacy level, thus invalidating the concept of cloaking area itself.

Definition 2.1. Consider a user u issuing a query with privacy level k. A *cloaking area* (built by the LTS) is any area A such that a set of users AS (called, *anonymity set*) can be found by the LTS within A such that $u \in AS$ and $|AS| \geq k$.

As earlier explained, a cloaking area so defined is not enough to guarantee the required privacy level, even though users inside AS are really indistinguishable for the adversary. Indeed, a technique only satisfying the above requirements is vulnerable to reverse-engineering attacks [11].

To prevent this problem, the following property has to be required.

Definition 2.2 (*Reciprocity Property [10]*)**.** Consider a user u issuing a query with privacy level k, associated by the LTS with a cloaking area A, and involving the anonymity set of users AS. We say that A satisfies *reciprocity* if every user in AS also generates the same anonymity set AS for the given privacy level k. A cloaking-area-construction algorithm is *reciprocal*, if every returned cloaking area A (for each possible user, and each possible privacy level k) satisfies reciprocity.

Clearly, the condition stated in Definition 2.2 is also satisfied if every user in AS also generates the same cloaking area A (which then involves the same anonymity set AS). As a matter of fact, this is the condition satisfied by reciprocal algorithms proposed in the literature, starting from the first proposal given in [10]. On the other hand, different returned cloaking areas, even including the same AS, could enable reverse-engineering attacks if not adequately built. From Definition 2.1, immediately follows that a cloaking area satisfying reciprocity satisfies also location k-anonymity. However, Definition 2.2 entails that k-anonymity ensures equal probability $\frac{1}{|AS|}$ for the adversary to identify a user. As shown in [10,11], without reciprocity, reverse-engineering attacks are possible allowing the adversary to reduce the above probability below the aimed level $\frac{1}{k}$. Note that reciprocity has been independently formulated as the *k-shared* property in [11].

3. Related work

In this section, we contextualize our proposal in the state of the art. We structured this section into three subsections. In the first subsection, we review the literature related to the subject of our work by following a proper classification. In the second subsection, we describe in detail a technique, called Casper [29], which we used as underlying technique to experimentally validate our method in Section 9. In the last subsection, we illustrate the advancements included in this paper with respect to a previous paper presenting preliminary results [30].

3.1. Survey on the related literature

Our paper is related to the wide literature existing in the field of privacy-preserving LBS. In this field, different types of protection techniques are available in the literature depending on the asset to be protected, as described in [1,27,31,32]. In detail, it is possible to categorize the LBS services, according to the asset to protect, into *user's location*, *query content* and *user's identity*.

User's location. In the case of user's location, the goal is to protect the location information through a perturbation on location data or by hiding realistic data within dummy information [33–36].

In [33], the authors propose an approach to retrieve POIs by iteratively performing requests starting from a fake location.

[34] focuses on providing an algorithm to generate a set of realistic dummy locations to achieve k-anonymity.

In [37], the authors emphasize the importance of decentralized approaches, in which secret sharing is used to obtain position obfuscation. The proposed techniques aim to ensure, however, a good quality of the service despite the obfuscation allowing contrast also inference attacks. A specific focus on inference attacks is given in [38]. In particular, in [38], the authors consider the semantic information behind the location of the real road network i.e., the solution protects the user's location and his query location simultaneously.

A recent paper that shares similarities with our work is [39], in which the authors apply an edge paradigm to provide LBS. However, the edge paradigm is not employed to obtain a decentralized architecture.

[40] deals with a crowdsourcing scenario in which many workers cooperate to provide accurate locations to the server. The paper focuses on task allocation and takes into account privacy aspects by employing obfuscation techniques.

A type of perturbation used to obtain obfuscation is enlarging, adopted in many works available in the literature [41,42].

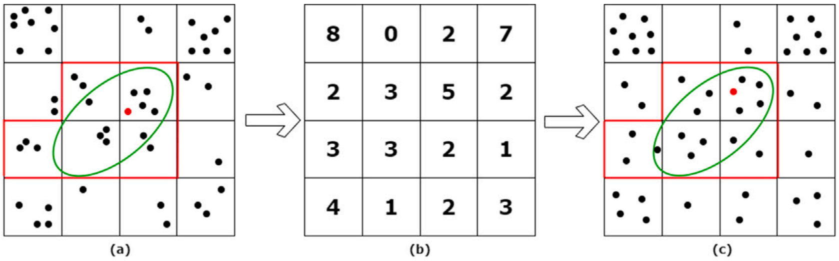


Fig. 1. An example of construction of an approximate cloaking area.

Specifically, in [41], a cloaked location is an uncertainty region which satisfies the privacy constraints specified by the user in the privacy profile. In [42], the authors use enlarging combined with other *obfuscation operators* to protect the user's location.

Query content. Query content techniques have the purpose of protecting the content of LBS queries. There exist many proposals that belongs to this category [43,44].

[43] proposes a novel query-perturbation-based scheme that protects query privacy in continuous LBS even when user identities are revealed.

In the framework proposed in [44], for each query issued by a mobile user, the client will construct a group of dummy queries and then submit them together with the user query to the server. This approach makes it challenging for the untrusted server to identify the specific user query.

Also this approach is prone to inference attacks.

Differential-privacy or machine-learning-based techniques [17,45–47] exist to increase the robustness of obfuscation and perturbation techniques against inference attacks.

In [45], the authors propose a differential-privacy-based approach to protect user location, that takes into account the temporal correlations in location data.

[46], through a GAN approach [48], produces noise that perturbes location data.

To address Long-Term Observation Attacks, [47] combines k-anonymity and differential privacy.

In [17], the authors describe a general probabilistic model for obfuscation mechanisms for the locations and measure its utility with respect to an arbitrary loss function.

Other privacy-preserving approaches are based on Private Information Retrieval (PIR) [49]. PIR protocols are based on cryptographic techniques for retrieving an item in a database without revealing the item.

User's identity. The last category of techniques is the user's identity, which is the category our paper belongs to. In this case, by making sufficiently anonymous the user, the sensitivity of the query does not threaten their privacy. In the literature, the approach well-consolidated aimed to protect the user's identity is based on *cloaking areas* which are zones that include at least k users and allow us to achieve *location k-anonymity* [8]. Some very recent papers that use this approach are [20,50–52].

[26] applies the k-anonymity property to vehicular networks (see next for more details). [50] builds an irregular polygon region. Then, k-anonymity is achieved by enlarging this region or adding fake positions.

[51] uses obfuscation and k-anonymity to build cloaking areas in an infeasible region (i.e., a region such that an entity cannot possibly be physically present at that location).

[20] relies on the blockchain to provide a trust mechanism for the construction of a cloaking area.

Another application of location k-anonymity (but with a different goal) is provided in [53] in which the authors face the problem of malicious users providing wrong coordinates to the LTS provider.

Our paper falls within this type of technique, and, in particular, in those reaching this goal by using a Trusted Third Party, called Location-Trusted Service (LTS) [8,54].

[8] introduces the concept of *location k-anonymity* and proposes a cloaking-area construction algorithm based on quad-tree.

[54] employs an entity, termed Function Generator, to distribute the spatial transformation parameters periodically, with which the users and the LSP can perform the mutual transformation between a real location and a pseudo-location.

Location k-anonymity is explained in detail in Section 2. To have effective location k-anonymity, it is necessary to satisfy *reciprocity* [10,11] that is fulfilled if the returned anonymity set includes at least k users whose anonymity set is the same. Considerable effort, including recent contributions, has been devoted in the literature [55,56] to obtain cloaking areas with robust privacy guarantees. In particular, [55] employs a GAN to construct a cloaking area, while [56] proposes a cloaking area construction algorithm designed to minimize waiting time in the continuous location-based services model.

Another class of techniques aimed at obtaining cloaking areas is based on *density services* [57,58]. In this case, no trusted party is involved (thus, the LTS does not exist). The users, in collaboration, relying on public services making available the distribution of people in the territory, construct autonomously the cloaking area. Then, through an anonymous communication network, submit the query along with the cloaking area directly to the LBS provider. Filtering the response is also in charge of the users. We observe that the density-service-based system proposed in [58] is hierarchical too. Anyway, this class of techniques is not comparable with the LTS-based techniques, for evident reasons.

Despite the fact that a number of LTS-based hierarchical approaches exist in the literature [59,60], this aspect is restricted to the cloaking-area-construction algorithm. Both works [59,60] propose a centralized hierarchical cloaking area algorithm satisfying reciprocity.

In our case, the hierarchy is at the basis of a multi-organization distributed implementation of the LTS system. To the best of our knowledge, the only existing proposal of a distributed LTS system is given in [61]. However, there are many differences with our approach.

First, the solution proposed in [61] is not hierarchical and requires the presence of a *secure comparison server* and a *directory server*. Furthermore, the user has to actively participate in the protocol by encrypting some data. Another difference is that even though the authors enable location broker overlapping, each user can register only with a single broker. On the other hand, in our approach, we allow users to register with multiple LTSs (playing the role of location broker) and this enables a practical business model (see Section 7). Finally, in [61], no explicit mechanism is provided to protect the anonymity of the communication against a global adversary able to observe the flow of exchanged messages.

Concerning the global adversary threat model, the only existing approach is based on P2P collaboration among users to mix LBS requests and prevent traffic analysis ([62] is a paper well representing this approach). Other P2P approaches to resolve the single-point-of-failure problem of the LTS have been proposed [63,64]. [63] is a P2P approach leveraging a distributed hash table to build cloaking areas including k users. Instead, [64] proposes a decentralized location privacy protection scheme based on obfuscation.

However, in our paper, we discuss scenarios in which P2P collaborative approaches may be considered less realistic, as users are generally not inclined to open their devices to incoming anonymous traffic. Furthermore, the incentive mechanism is not effective. The only non-P2P approaches in previous works involve a Trusted Third Party (TTP) [54,65] (referred to as *Function Generator* and *Converter*, respectively).

While these approaches could, in principle, be combined with our technique, the authors themselves emphasize a significant drawback: both the LTS and these additional TTPs are not permitted to collude with malicious users. This consideration overlooks the fact that any entity can always assume the role of a user.

Vehicular networks. We reserve a separate class for privacy solutions in vehicular networking that inherently uses location information. However, vehicular services are a particular case of continuous services. Some interesting surveys related to this aspect can be found in [66,67].

[68] does not deal with privacy aspects but proposes an architecture for location-based trustworthy service recommendations.

[69] applies differential privacy to prevent location perturbation behavior of users from being recognized by guaranteeing their pseudo-locations plausible.

[70,71] propose a Security Credential Management System (SCMS) including a Location Obscuring Proxy. However, the proposed approaches regard a privacy-preserving way to manage and distribute cryptographic credentials to the vehicles and do not provide information about the provision of location-based services.

The scenario and threat model are deeply different from ours. Indeed, we assume that the location of the users is known by the attacker (LBS provider). Our goal is to prevent an LBS request (a concept not discussed in [70,71]) from being linked with the identity of the user performing the request. In contrast, the objective of [70,71]) is to prevent vehicle tracking.

[72] focuses on the provision of conditional pseudo-anonymity in which accountability (i.e., the possibility to link the identity of users with their pseudonyms) should be provided along with pseudo-anonymity. This is different from our objective i.e., identity protection through location k-anonymity. Moreover, the paper does not discuss the management of location information but only how to map pseudonyms with the real identity of the vehicles.

Some proposals adopt blockchain technology [26,73,74] in vehicular networks.

In [26], the authors propose a location privacy protection method based on double k-anonymity that hides user locations and requests information. The cloud server is introduced as a trusted third party to isolate the direct communication between users and the service provider, while the correlation between identities and requests is also reduced by means of a permutation and combination method.

[73] aims to obtain privacy of location services in vehicular networks. However, this solution is weakly related to our paper because there is no LBS provider and the location services are represented by information shared by the users.

Similarly, [74] does not propose privacy techniques but focuses on the trust of shared location information through blockchain.

In [75], the role of LTS is played by the roadside unit (RSU) and achieves both query and location privacy.

[76] propose a cloaking area construction algorithm based on obfuscation to protect users' location.

A general perspective is provided in [77]. Therein, the authors proposed a formal framework for the analysis of Location Privacy-Preserving Mechanisms (LPPM) in navigation services and applied this framework to evaluate two standard approaches (i.e., k-anonymity and differential privacy).

Comparison. To conclude this section, we summarize the main properties of the investigated solutions in Table 1. Specifically, Column 2 represents the entity to protect according to the standard categorization (*user's location*, *query content*, and *user's identity*). Column 3 describes if the proposal is adopted for searching POI or for continuous LBS. In Column 4, we report the privacy techniques used in the proposal. Column 5 defines if the proposal offers protection against a global adversary (GA) able to observe the entire traffic exchanged by the involved actors. Column 6 reports the adopted architecture (decentralized or centralized). Finally, in Column 7, we report the data source used to test the proposal.

We observe that, besides our paper, only [62] achieves resistance against the global observer. However, [62] does not include an LTS. This fact, together with all the arguments given in the introduction, shows the advancement of our paper with respect to the state of the art.

3.2. The Casper algorithm

We dedicate a subsection of the related work to describe Casper [29], a state-of-the-art algorithm for cloaking-area construction. Since our approach is parametric with respect to any cloaking-area-construction algorithm, we use Casper in the experimental validation to measure the performance of our proposal.

Casper is an LTS-based system aimed to achieve location k-anonymity. The Casper system architecture consists of two components: the *location anonymizer* and the *privacy-aware query processor*.

The location anonymizer assumes the role of the LTS and is responsible for constructing cloaking areas. During registration, users set a *privacy profile*, defined as a pair (k, A_{min}). Here, k denotes the privacy level, while A_{min} represents the lower bound for the size of the returned cloaking area (as overly small cloaking areas can pose a threat to users' privacy). Users have the flexibility to modify their privacy profile at any time.

The anonymizer maintains the locations of the clients using a pyramid data structure, similar to a Quad-tree, where the minimum cell size corresponds to the anonymity resolution. The privacy-aware query processor is embedded inside the LBS provider. It handles anonymous queries and cloaking areas. The privacy-aware query processor returns a list of candidate answers to the location-based query via the location anonymizer. The size of this candidate list is significantly influenced by the user's privacy profile; for instance, a stringent privacy profile would yield a larger list of candidates. Casper is set in our experiments with a very high granularity level (indeed, as mentioned earlier, it allows granularity modulation), in such a way that negligible approximation is introduced by Casper itself.

As a final remark, we highlight that Casper is not comparable with our approach since it does not achieve the same goal, i.e., the implementation of a distributed hierarchical LTS system.

While Casper implements a cloaking-area algorithm, our approach leverages any existing algorithm (Casper itself as an example) to return an approximate cloaking area computed in a distributed hierarchical LTS system. Without our technique, the hierarchical LTS system could not directly leverage any existing cloaking-area algorithm. In other words, our technique can be viewed as a way to enable the distributed LTS approach, and then to obtain the advantages that the paper tries to highlight.

3.3. Advancements with respect to previous results

The present paper is a wide extension of a short paper presented in [30]. Although the rough original idea underlying the present research is included in [30], there are a number of relevant novel contributions provided in this article. First, we conduct an experimental campaign (missing in [30]) aimed to validate our proposal both from the effectiveness and the network performance point of view. Second,

Table 1
Comparative table.

Proposal	Protected entity	LBS Type	Technique	GA resistance	System architecture	Data source
[8]	identity	POI	k-anonymity	no	centralized	USGS dataset
[29]	identity	POI	$k-$anonymity	no	centralized	Brinkhoff Generator [78]
[17]	location and identity	POI	differential privacy	no	decentralized	Not Available
[26]	identity	continuous	k-anonymity	no	centralized	SUMO simulator
[20]	identity	continuous	k-anonymity	no	decentralized	Self generated
[33]	location	POI	obfuscation	no	centralized	Tiger dataset
[34]	location and identity	POI	k-anonymity and obfuscation	no	decentralized	Levy walk model in NY city
[37]	location	POI	obfuscation and Secret Sharing	no	decentralized	Location Sharing Services Dataset
[38]	location and identity	POI	k-anonymity and machine learning	no	centralized	Brinkhoff Generator [78]
[39]	location	POI	obfuscation	no	centralized	Self generated
[40]	location	POI	obfuscation	no	decentralized	Geolife dataset
[41]	location	POI	obfuscation (enlarging)	no	centralized	Self generated
[42]	location	POI	obfuscation (enlarging)	no	centralized	Self generated
[43]	query content	continuous	obfuscation	no	decentralized	TIGER dataset
[44]	query content	POI	obfuscation	no	decentralized	Gowalla and SNAP datasets
[45]	location	POI	differential privacy	no	decentralized	Geolife and Gowalla datasets
[46]	location	POI	machine learning (GAN)	no	decentralized	Gowalla dataset
[47]	location and identity	POI	k-anonymity and differential privacy	no	decentralized	Brightkite and Gowalla datasets
[50]	identity	POI	k-anonymity and obfuscation	no	centralized	Brinkhoff Generator [78]
[51]	identity	POI	k-anonymity and obfuscation	no	centralized	Self generated
[54]	identity	POI	$k-$anonymity	no	centralized	Self generated
[55]	identity	POI	machine learning (GAN)	no	centralized	Competition System of Peeking University
[56]	identity	continuous	$k-$anonymity	no	centralized	Geolife dataset
[57]	identity	POI	k-anonymity	no	decentralized	Milano dataset
[58]	identity	POI	k-anonymity	no	decentralized	Milano dataset
[59]	identity	POI	k-anonymity	no	centralized	R-Tree Portal
[60]	identity	POI	k-anonymity	no	centralized	Self generated
[61]	identity	POI	k-anonymity	no	centralized	Self generated
[62]	identity	POI	k-anonymity	yes	decentralized	Not available
[63]	identity	POI	k-anonymity	no	decentralized	Brinkhoff Generator [78]
[64]	identity	POI	k-anonymity	no	decentralized	Self-generated
[65]	identity	POI	k-anonymity	no	decentralized	Brinkhoff Generator [78]
[69]	location	continuous	differential privacy	no	decentralized	Geolife data set
[75]	location and query content	continuous	obfuscation	no	centralized	Self generated
[76]	location	continuous	obfuscation	no	centralized	Open Street Map
This paper	identity	POI	k-anonymity	yes	decentralized	Brinkhoff Generator [78]

we extend the basic system model by introducing multiple services and LTS competence overlap. This extension is not trivial, as it can be realized by going in-depth in Section 6, because it affects both the hierarchy topology and the position-notification mechanism, which becomes much more tricky and needs some theoretical new results. Another contribution not included in [30] is an evaluation of the computational complexity of our hierarchical approach.

Moreover, a plausible business model is described in this paper, with the aim to increase the concreteness of our proposal, starting from the observation that the distributed LTS system may open new markets allowing the entry of multiple organizations, possibly SMEs. This is a relevant point, because, so far, the market of location-based services seems destined to stay confined to the domain of huge international players. Finally, we trace the road towards an implementation of our system exploiting the edge–cloud paradigm, which appears compliant with both the hierarchical distributed system architecture and the local (multicast) communication features. The experiments validate that the above direction is convincing.

4. Approximate cloaking areas

In this section, we introduce our method to generate approximate cloaking areas. First, recall that our method relies on any existing algorithm for constructing reciprocal cloaking areas (refer to Definition 2.2). It is easy to see that, when considering a specific privacy level k, the property of reciprocity implies location k-anonymity. We recall that an algorithm is reciprocal if the returned cloaking area encompasses a minimum of k users and remains the same when the algorithm is invoked by all these users. As discussed in Section 2, our paper follows the notion of reciprocity defined in the existing literature, as stated in Definition 2.2, wherein the same cloaking area is returned to each user within the anonymity set AS. Furthermore, throughout the paper, we assume implicitly that the anonymity set comprises all users belonging to the given cloaking area.

We are now ready to describe our approach. The method works on a geographical area, say A, which consists of a set of *positions*. Each position, as usual, corresponds to the classical GPS coordinates. Consider that, even though an area of the territory is an infinite compact set of positions, for our purpose, it does not matter to represent areas in a strict sense. Indeed, we are interested only in those positions associated with users. We denote the set of positions associated with all users within area A as $P \subseteq A$, which is referred to as the *actual position set*.

Our paper does not propose a new cloaking-area-construction algorithm but provides a method to achieve approximate cloaking areas starting from any existing cloaking-area-construction algorithm. We formalize this aspect through the notion of *Cloaking Oracle*, which represents *any* cloaking-area-construction algorithm. We highlight that we require, the oracle, to be reciprocal.

Definition 4.1. A *Cloaking Oracle $\mathcal{O}_A$ (for the area A)* is a partial function that receives as input a privacy requirement k, a set of positions Q, and a position $p \in Q$, and returns, if any, a *cloaking area $C_Q \subseteq A$* such that $p \in C_Q$ and satisfies k-reciprocity.

According to the definition of reciprocity recalled in Section 2, from Definition 4.1 it follows that there exist at least k positions $p_1, \dots, p_k \in C_Q \cap Q$ including p such that $\mathcal{O}_A(k, Q, p_1) = \dots = \mathcal{O}_A(k, Q, p_k) = C_Q$.

The Cloaking Oracle is a partial function because the existence of a k-reciprocal cloaking area for each input is not guaranteed (for example, if $|Q| < k$, no cloaking area can be built). But, as observed earlier, we can consider as a Cloaking Oracle, an *ideal* algorithm. Therefore, this aspect does not refer to a limit of our work, but only to the intrinsic possible non-compliance of the distribution Q with the privacy requirement.

Now, consider a partition Z_A of A identifying sub-areas of the territory. Each element of Z_A is called *cell*. As it will be clear in the following, our method could work for any shape of the cells. However, for the sake of simplicity we choose, in Section 5, to consider equi-sized squares as cells.

Now, we introduce the following two definitions. The former introduces a function that counts the number of users that belong to each cell. The latter introduces a way to classify (into equivalence classes) distributions of users in such a way that two distributions are equivalent if preserve the same number of users per cell.

Definition 4.2. The *aggregation mapping* (*on P*) is a function $D_P : Z_A \to \mathbb{N}$ which, for each cell, returns the number of positions of P in it included. Formally, $D_P(z) = |z \cap P|$, for each $z \in Z_A$.

Definition 4.3. Given D_P, a set of positions $P' \subseteq A$ is said *equivalent* to P, if $D_P = D_{P'}$(i.e., $|z \cap P| = |z \cap P'|$, for each $z \in Z_A$).

We denote by $\mathcal{R}(D_P)$ a given fixed *generation schema* (i.e., a deterministic function) of a set of positions equivalent to P. Therefore, given D_P, any party (i.e., the various LTSs) that is aware of this schema can deterministically generate the same set of positions $P' = \mathcal{R}(D_P)$ equivalent to P.

A straightforward way to implement $\mathcal{R}$ is as follows: the actual positions in a cell are ordered according to their coordinates (e.g., using coordinate-wise order). Then, for an equivalent set of positions, we consider the equidistributed sequence of points in the cell. Each actual position is associated with the nearest position in the set of equivalent positions.

The intuitive role of function $\mathcal{R}$ is to enable the construction of a k-reciprocal cloaking area just by calling the Cloaking Oracle on a set of positions P' equivalent to P. This will be clearer in the following when the approximate-cloaking-area construction algorithm will be described. Intuitively, the idea is that such an algorithm will be used by LTSs organized in a hierarchy. While the LTS of the lowest level (i.e., level 0) can use directly the actual positions, higher-level LTSs do not know this set, but just the aggregation mapping, i.e., the number of users per cell. Then, through $\mathcal{R}$ they are able to compute an equivalent set of positions without knowing the actual set of positions.

Now, we introduce the following notations, adopted from now in the paper:

- By A, we denote the given geographical area,
- by P, we denote the given actual position set,
- by Z_A we denote the given partition of A, and
- by $\mathcal{R}$, we denote the given generation schema.

In the following definition, we introduce the concept of the *approximate cloaking area*. Informally, for a given cloaking area, its approximate cloaking area is defined as the union of cells that are (also partially) overlapped by the original cloaking area.

Definition 4.4. Given a cloaking area C_Q, for a certain set of positions Q, we denote by $S_{C_Q} = \{z \in Z_A | z \cap C_Q \neq \emptyset\}$. We define the *approximate cloaking area (of C_Q)* as $\overline{C}_Q = \bigcup_{z \in S_{C_Q}} z$.

S_{C_Q} represents the set of cells that are involved by C_Q and that $\overline{C}_Q$ identifies a sub-area of A including C_Q.

We are ready to define the algorithm that allows us to build a k-reciprocal clocking area by relying only on the aggregation mapping D_P, instead of the actual set of positions P, by exploiting the generation schema $\mathcal{R}$.

We denote the Algorithm 1 by $\mathcal{A}_{cloak}(k, D_P, p')$ and call it *approximate cloaking-area construction*.

$\mathcal{A}_{cloak}(k, D_P, p')$ receives as input:

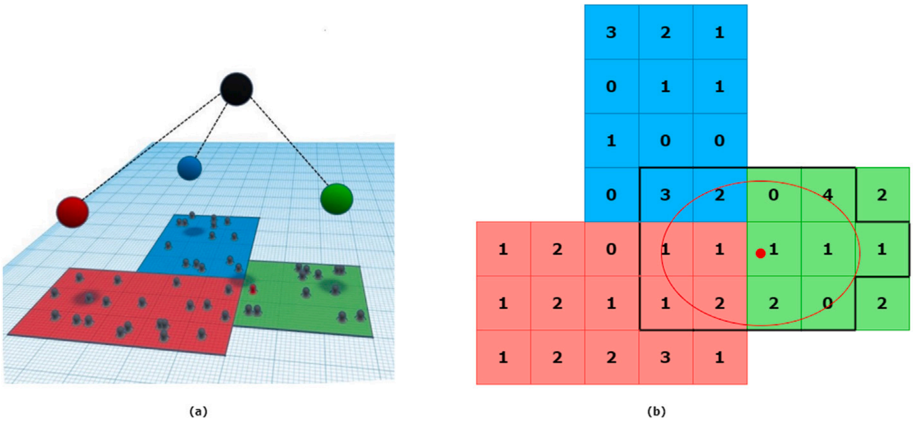
- the privacy requirement k,
- an aggregation mapping D_P,
- $p' \in \mathcal{R}(D_P)$

and outputs either *fail* or an approximate cloaking area.

$\mathcal{A}_{cloak}$ proceeds as follows:

1. Relying on D_P and $\mathcal{R}$, it builds the position set $P' = \mathcal{R}(D_P)$ (recall that P' is equivalent to P);
2. It calls $\mathcal{O}_A(k, P', p')$. If the Oracle is not able to respond (i.e., $\mathcal{O}_A(k, P', p')$ is not defined), then $\mathcal{A}_{cloak}$ returns *fail*, else ($C_{P'} = \mathcal{O}_A(k, P', p')$), the execution goes to the next step;
3. $\mathcal{A}_{cloak}$ returns the approximate cloaking area $\overline{C}_{P'}$ of $C_{P'}$.

In Fig. 1, we report an example of execution of the approximate cloaking-area construction.



(a) (b)

Fig. 2. Example of LTS hierarchy and approximate cloaking area for the LTS of level 1.

Algorithm 1 Approximate cloaking-area construction

```
Notation  $\mathcal{O}_A$: the Cloaking Oracle for the area $A$
Notation  $\mathcal{R}$: the generation schema
Input  $k$: privacy requirement
Input  $D_P$: aggregation mapping
Input  $p'$: a position in $\mathcal{R}(D_P)$
Output  fail or $\overline{C}_{P'}$
$P' = \mathcal{R}(D_P)$
if ($\mathcal{O}_A(k, P', p')$ is not defined) then
    return fail
else
    $C_{P'} = \mathcal{O}_A(k, P', p')$
    return $\overline{C}_{P'}$ of $C_{P'}$
end if
```

Specifically, map **(a)** represents the geographical area A divided into cells by Z_A, including a certain set of users with actual positions P, and a given position (in red), which the LBS request comes from. To understand the meaning of the red and green areas, we need first to describe the other maps. Suppose that the privacy requirement is $k = 12$ so that the final cloaking area must include at least 12 users. Map **(b)** denotes the application of the aggregation function on P. Finally, map (c) shows the redistribution $P' = \mathcal{R}(D_P)$ of users (preserving the aggregation mapping), the cloaking area $C_{P'}$ (in green) returned by the Cloaking Oracle, and the approximate cloaking area $\overline{C}_{P'}$ (in red). The example shows that even though the Cloaking Oracle returns on P' the green cloaking area fulfilling the privacy requirement (as it includes k=12 positions), being P' a set of dummy positions, the cloaking area itself could be not valid if applied to the actual position set P. This is the case we are representing. Indeed, when projecting the green cloaking area from map (c) to map (a), we see that the actual involved positions are fewer than the required number (as they are 10 positions).

In contrast, the red approximate cloaking area, which, by construction, includes the same number of positions if applied to either P or P' (for P and P' equivalent), satisfies the privacy requirement as the number of actual (or dummy) included positions equal to 16.

For simplicity, in this example, we did not consider the reciprocity property.

5. The distributed LTS

In this section, we describe the LTS organization. The area A is divided into n rectangles each containing a number of square cells in Z_A. Let $A_Z = \{A_1, \dots, A_n\}$ denote the set of such rectangles. The choice of the rectangle as the shape of an elementary group of cells is done only for the sake of simplicity. Any other shape could be utilized in principle.

In our system, multiple hierarchical LTSs are adopted, each possibly managed by an autonomous organization. They are organized as follows. Each rectangle, called 0-zone, is under the responsibility of an LTS of level 0. Each LTS of level 0 responsible for the 0-zone A_i (with $1 \le i \le n$) knows a subset of actual positions $P_i \subseteq P$ representing the actual positions of the users in the 0-zone A_i.

The LTSs system basically implements a forest of tree spatial indices. Specifically, a number of LTSs of level 0 managing adjacent 0-zones constitutes the set of children of an LTS of level 1. Such an LTS is responsible for a 1-zone obtained as the union of the 0-zones which each of its children is responsible for and knows only the restriction of the aggregation mapping D_P to the subset of P involved by the 0-zones forming the 1-zone of its competence.

In general, a number of LTSs of level i managing adjacent i-zones constitutes the set of children of an LTS of level $i + 1$. Such an LTS is responsible for an $(i+1)$-zone obtained as the union of the i-zones which each of its children is responsible for. It knows only the restriction of the aggregation mapping D_P to the subset of P involved by the 0-zones mapped to the $(i + 1)$-zone of its competence. In other words, each LTS of level $i > 0$ has competence on a number of rectangles of A and, then, on the involved cells but knows only, for each cell, the number of users positioned in it and not the exact position. This happens also for the root of each tree, which is set in such a way that the size of the map of its competence is feasible. This is why we have not a single tree but a forest of trees. We assume that all the LTSs share the generation schema $\mathcal{R}$.

The LTS hierarchical organization is sketched in Fig. 2.(a). In detail, there are three 0-zones, colored in green, red, and blue, respectively, and their LTSs of level 0. Users are also represented. The red user is submitting an LBS query. The union of these three 0-zones forms the 1-zone of competence of the black LTS of level 1. Fig. 2.(b) reports the view from the black LTS of this 1-zone (which includes only the aggregation mapping). The red dot corresponds to the position which the query comes from. The red-line ellipse and the black-line region will be described later.

5.1. Registration

When a user enters a 0-zone, it performs a pseudonymous registration to the LTS of level 0 responsible for such a zone.

Each LTS of level $i \ge 0$ (in a tree of the forest) knows its parent LTS (of level $i + 1$) according to the LTS hierarchy defined above. Similarly, each LTS of level $i \ge 1$ knows all its children (LTS of level $i - 1$).

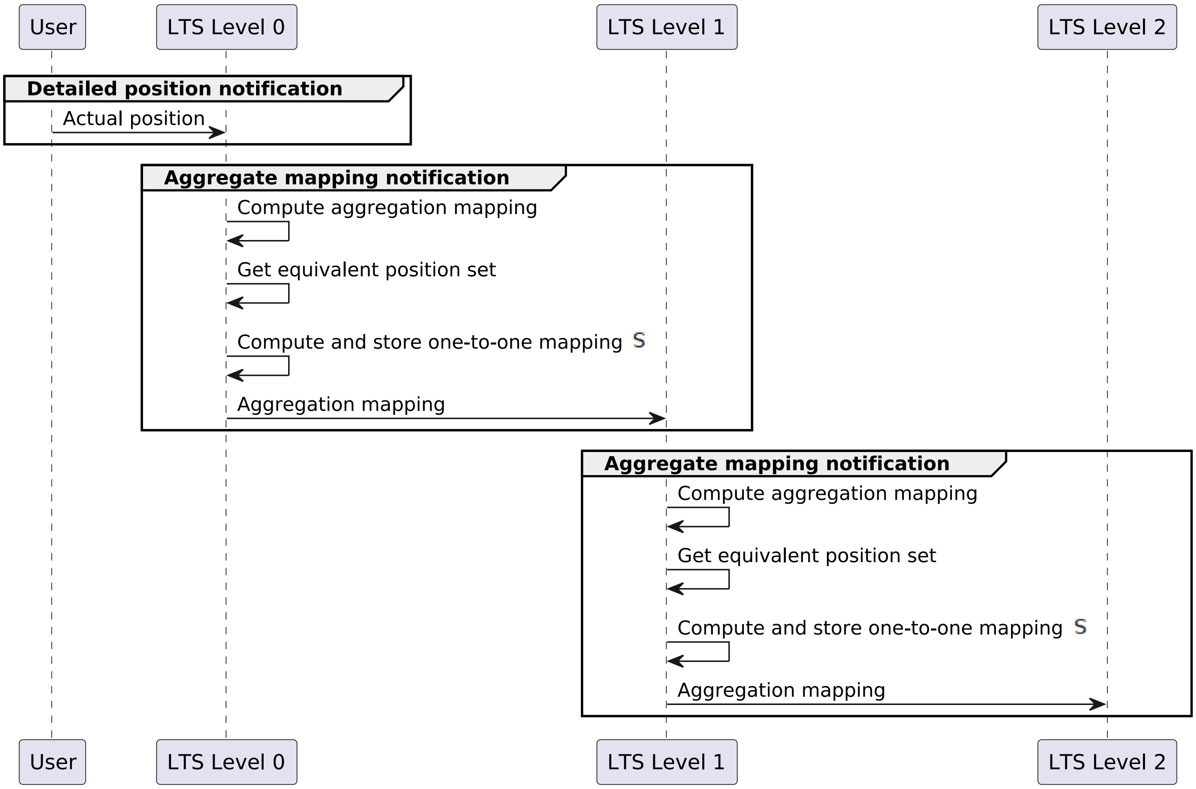

User
LTS Level 0
LTS Level 1
LTS Level 2
Detailed position notification
Actual position
Aggregate mapping notification
Compute aggregation mapping
Get equivalent position set
Compute and store one-to-one mapping S
Aggregation mapping
Aggregate mapping notification
Compute aggregation mapping
Get equivalent position set
Compute and store one-to-one mapping S
Aggregation mapping

Fig. 3. Position notification.

5.2. Position notification

Periodically, with a given frequency, each user located in a 0-zone A_i sends their position to the LTS of level 0 responsible for that 0-zone. Therefore, such LTS, say L, knows the set of positions P_i of the users in the 0-zone. This step of the position notification process is called *detailed position notification*.

Now, we describe how aggregate positions are notified to the higher levels of the LTS hierarchy. We call this task *aggregate mapping notification*. Let us consider the LTS L again. We denote by D_{P_i}, the restriction of D_P to P_i. L can compute $P'_i = \mathcal{R}(D_{P_i})$. Then, it builds and maintains a one-to-one correspondence $\mathcal{S}$ between the sets P_i and P'_i in such a way that any actual position $p \in P_i$ is associated with a dummy position $\mathcal{S}(p) \in P'_i$ such that $p, \mathcal{S}(p) \in z$ for some $z \in Z_A$ (i.e., p and $\mathcal{S}(p)$ are in the same cell). As we will clarify in Section 8, this is done to guarantee k-reciprocity (not with the purpose of hiding the actual position).

Periodically, L sends D_{P_i} to its parent LTS (of level 1). This counts the number of users for each cell contained in its 0-zone. At this point, the LTS of level 1, by merging the information coming from its children, is able to know the restriction of D_P to P^*, where P^* is the set of the positions of the users in the 1-zone which such an LTS is responsible for. Note that it does not know P^*. Periodically, this restriction of D_P is sent to its parent LTS (of level 2). The process is iterated until the root of the tree is reached.

The position notification procedure is sketched in the sequence diagram of Fig. 3.

5.3. LBS request processing

Suppose a user in position p performs an LBS request. To prevent the global adversary from identifying that a user submits a query to the LBS provider, the user replaces one of the messages of the detailed position notification (intended for the LTS of level 0 responsible for the 0-zone which the user is located in) with the LBS request including the position p and an on-the-fly key K. Suppose that such 0-zone is A_i and the set of positions of the users including in it is P_i, this LTS invokes the Cloaking Oracle $\mathcal{O}_A(k, P_i, p)$.

We distinguish 2 cases. The first case is when the Cloaking Oracle outputs a cloaking area C_{P_i}. In this case, the LTS can directly perform the request to the LBS provider. After receiving the response and (possibly) filtering it, the LTS sends the response encrypted with K in multicast to the users included in the cloaking area.

The second case occurs when the Cloaking Oracle fails. If this happens, the LTS of level 0 forwards the request to its parent LTS (of level 1), replacing p with $p' = \mathcal{S}(p)$.

Again, to conceal the transmission of a request, the LTS of level 0 substitutes one of the messages in the aggregate mapping notification (intended for its parent) with the request itself. At this point, the LTS of level 1 invokes $\mathcal{A}_{cloak}$ by passing as input the privacy requirement k, the restriction of the aggregation mapping to the positions set included in the 1-zone of its competence, and the position p'.

Since $\mathcal{A}_{cloak}$ might positively respond or fail, we have to distinguish two cases again. If $\mathcal{A}_{cloak}$ succeeds, then it returns an approximate cloaking area and the LTS of level 1 performs the request to the LBS provider. When it receives the response, filters it, and starts what we define as the *cloaking multicast* mechanism. This mechanism first requires the selection of the children LTSs of level 0 which the filtered response has to be sent to. The selection is done by finding the LTSs of level 0 which are responsible for at least one cell included in the approximate cloaking area. Then, the filtered response is multicasted to such LTSs encrypted with K. In turn, each LTS sends the response in multicast to all the users belonging to the involved cells.

Otherwise (i.e, $\mathcal{A}_{cloak}$ fails), the request is forwarded (by replacing, as usual, an aggregate-mapping-notification message) by the LTS of level 1 to its parent. The process is iterated until the root of the tree. The LBS request can be satisfied only if, eventually, an LTS is able to positively respond.

We report, in Fig. 4, a sequence diagram that describes how the LBS request is processed, in the case in which the Cloaking Oracle fails for the LTS of level 0 and it is successfully for the LTS of level 1.

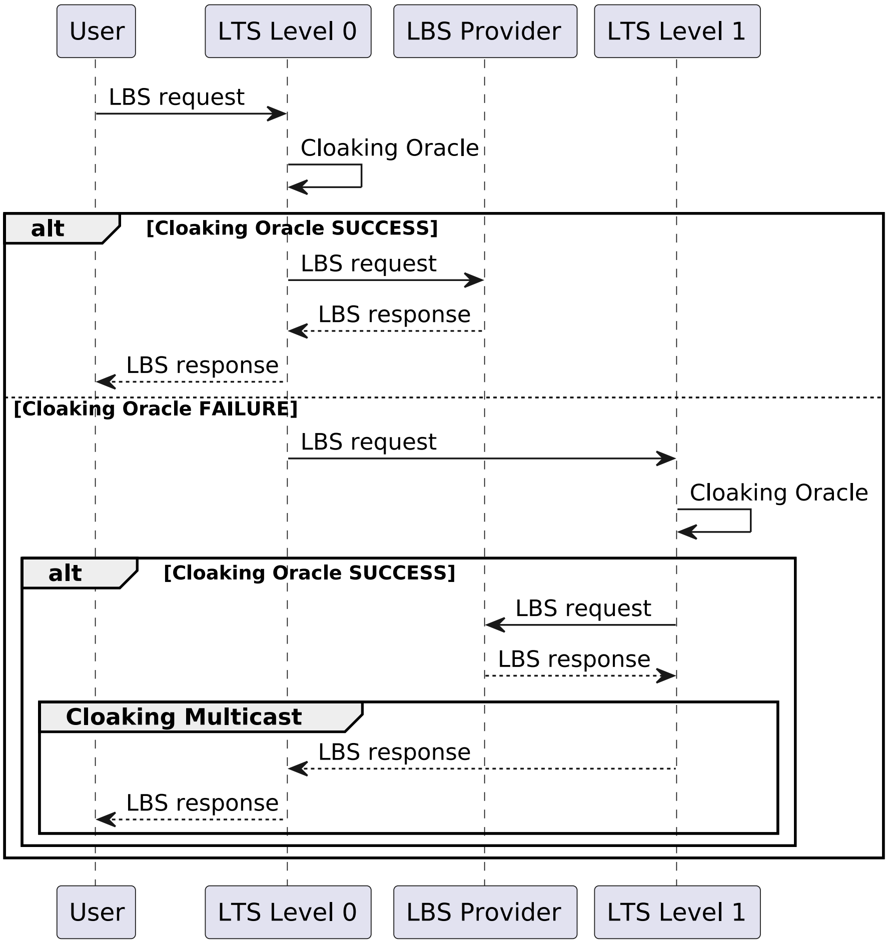


Fig. 4. LBS request.

In Fig. 2.(b), an example of an approximate cloaking area built by an LTS of level 1 is depicted. The red-line ellipse represents the cloaking area returned by the Cloaking Oracle invoked by the black LTS. The black-line region is the approximate cloaking area of the previous cloaking area.

6. Service management, LTS overlapping, and implementation aspects

The goal of this paper is to present a practical solution grounded in a theoretical framework for implementing a real-life LTS system. In this section, we introduce enhancements to the model to address practical considerations. Specifically, we introduce the dimension of the *service*, to allow an LTS to be involved only in some specific services from those available in the territory.

To provide more flexibility with respect to the basic model presented in Section 5, we allow also LTS competence overlapping, at any level of the hierarchy, thus moving from a forest of trees to a set of acyclic graphs.

We start by defining the role of the organizations involved in the system. The users belong to the set U. We assume that they are identified by their positions in P.

- An organization, called SP, providing the set of location-based services S_A in the area A.
- A set of organizations $L^i = \{L^i_1, \dots, L^i_{m_i}\}$ $(0 \le i \le i_{max})$ interested in providing the LTS services at level i. i_{max} represents the highest level of the LTS hierarchy (possible real-life values for i_{max} are 3 or 4) and m_i represents the cardinality of the set of organizations at level i.
- The telephone service provider TSP supporting communications.

An agreement between SP and TSP exists, to implement an edge–cloud solution sketched at the end of this section. Let ST be the entity representing this agreement. ST can play the role of a global passive adversary because it has the technological capabilities to monitor all the traffic generated and received by the users. Clearly, agreements between LTSs and TSP also exist to make edge–cloud communication effective. Indeed, queries are sent to LSTs and then forwarded to SP, and then also the communication between the user and the LTSs has to exploit the edge–cloud paradigm.

We define now a number of mappings associating services with 0-zones, LTSs, and users, and defining the LTS hierarchy.

- A mapping returning the set of services on which a given LTS of level 0 works in a certain 0-zone: $services\ :\ L^0 \times A_Z \rightarrow 2^{S_A}$. To define this mapping, every LTS of level 0 establishes which services it wants to support. A given LTS of level 0 keeps only the restriction of the function $services$ regarding itself. This allows us to store the information close to the user, coherently with the edge–cloud paradigm. Obviously, this information is notified to all the registered users.
- A mapping returning the set of LTSs of level 0 working on a given 0-zone: $zero\_lts\ :\ A_Z \rightarrow 2^{L^0}$. This mapping is managed by ST, and, again, notified to all the users occurring in a given 0-zone, concerning its restriction to this 0-zone.
- A mapping returning the set of LTSs of level $i+1$ with which a given LTS of level i $(0 \le i \le i_{max}-1)$ has established an agreement to forward users' requests: $up_i\ :\ L^i \rightarrow 2^{L^{i+1}}$, and a mapping

returning the set of LTS of level $i-1$ with which a given LTS of level i has established an agreement to receive users' requests: $down_i : L^i \to 2^{L^{i-1}}$.

- A mapping defining the set of services to which a user does not want to collaborate for the construction of the cloaking area: $deny\_s : U \to 2^{S_A}$. This mapping is defined by the user and is notified to and kept by the LTSs of level 0 to which the user is registered.
- A mapping returning, for a given LTS of level 0, say L, a service $s \in S_A$, and a 0-zone $A_j \in A_Z$, a set of positions P_j^L corresponding to users belonging to A_j of competence of L, registered with the LTS L, who did not deny the service s and $s \in services(L, A_j)$: $position : L^0 \times S_A \times A_Z \to 2^P$, where P, we recall, is the set of all the positions as defined in Section 4.

  The set P_j^L, referring to the 0-zone A_j, will be used by any LTS of level 0 L of competence of A_j, to build a cloaking area only for a query regarding the service s. Each LTS of level 0 keeps the proper restriction of this mapping, which is populated thanks to the detailed position notification described in Section 5.2.

  Given a cell $z \in Z_A$ included in A_j, we denote by $P_j^L[z]$, the subset of the positions of P_j^L belonging to the cell z.

Since the LTSs of level i may require the collaboration of LTSs of level $i+1$ to satisfy a request on a given service, we require that each LTS of level i (with $i > 0$), say L_x^i, supports all the services supported by the LTSs in $down_i(L_x^i)$.

The inclusion of multiple services and the changes introduced in the LTS hierarchy do not impact the mechanism by which the cloaking area is constructed, provided that this is done by selecting just the proper dimension (i.e., the service), to avoid intersection attacks. For example, suppose that in the 0-zones A_i the services s_1 and s_2 are provided, and, in the adjacent 0-zone A_j the services s_2 and s_3 are provided. Now, consider a user belonging to A_i who submits a query regarding the service s_1 that cannot be resolved by the LTS of level 0. If we allow the LTS of level 1 to use all positions (independently of the service) in A_i and A_j to build the cloaking area, then the adversary can infer that the actual anonymity set of users is restricted to A_i thus breaking k-anonymity. To avoid this, for a query on a service s, the positions that can be used for this service at any level of the hierarchy must come from 0-zones in which s is provided. This allows us, for the security analysis provided in Section 8, to consider the simpler model of a single service.

The introduction of the new features in the system model affects only the registration phase and the position notification.

6.1. Impact on the registration phase

The registration phase defined in Section 5.1 is affected by the inclusion of multiple services and LTS competence overlapping basically due to the mapping $deny_s$ and to the fact that the user can choose the LTSs to contact for any service.

Specifically, the registration is modified as follows. Each user performs a *global* registration (in pseudonymous form) with ST and obtains a *global* pseudonymous ID.

When a user requires the collaboration of an LTS of level 0 to obtain a service, they perform a *local* registration with such an LTS and provide it with their global pseudonym. Therefore, all the LTSs of level 0 to which the user is registered will own its global pseudonymous ID. As it will be clear in the next section, we need a unique identifier for the users because the higher-level LTSs should have a global view (even in aggregate form) of the user distribution. Therefore, users who are registered with multiple LTSs of a given 0-zone should be counted once, and then their records should be reconciled. The exact mechanism to obtain this goal without breaking users' privacy is explained in Section 6.2.

Given a user u belonging to a 0-zone A_j, u can perform the local registration with one or more LTSs of level 0 belonging to the set $zero\_lts(A_j)$.

For each LTS L_i^0 chosen by the u, we assume *by default* that u provides their positions to build cloaking areas for all the services in $services(L_i^0, A_j)$. Anyway, u can specify the services from which they choose not to adhere and communicate this preference to the selected LTSs of level 0. This operation represents the manner in which the function $deny$ is updated.

6.2. Impact on the position notification

The revision of the basic model provided in Section 6 has an impact on the position notification defined in Section 5.2. This basically happens due to the presence of multiple services and to LTSs competence overlapping.

The presence of multiple services induces a trivial change. Indeed, when a user denies a service, all the LTSs of level 0 in which the user is registered should be made aware of this denial. Therefore, the position of that user is not considered for that service. Moreover, the view of the positions of the users is specific for a certain service, as induced by the definition of the mapping $positions$ introduced earlier.

From now on, consider implicitly given a service s, and only those users who did not deny this service.

Concerning the detailed position notification, the only difference with respect to the basic definition is that the set of positions that is updated by an LTS of level 0, say L, working on a 0-zone A_j, is just the partial view $P_j^L = positions(L, s, A_j)$ of the overall set of positions of users belonging to A_j. Therefore, two LTSs A and B of level 0 working on the same 0-zone, may have a different view of the positions of the users occurring in this 0-zone. In other words, it may happen that $P_j^A \neq P_j^B$, because there could be a difference between the set of users registered with A and registered with B.

Also the aggregate mapping notification to the higher level is affected by the overlapping of LTSs of level 0. For instance, it might happen that on a given cell $z \in Z_A$, belonging to the 0-zone A_j, there are two LTS of level 0 of competence, say A and B. In this case, some users located in the cell z could be registered only with A, some users could be registered just with B, and other users could be registered with both A and B. Let denote by k_A (k_B, respectively) the number of users registered only with A (B, respectively) and let k_{AB} the number of users registered with both A and B. It happens that, the actual restriction of the aggregation mapping to notify to the proper LTSs of level 1 should be $k_A + k_B + k_{AB}$. Unfortunately, thanks to the detailed position notification, the only information available to A is $|P_j^A[z]| = k_A + k_{AB}$, whereas the information available to B is $|P_j^B[z]| = k_B + k_{AB}$. To overcome this drawback, the two LTSs should know the cardinality of the intersection between the $|P_j^A[z]|$ users registered with A and the $|P_j^B[z]|$ users registered with B. To avoid unwanted leakage of privacy, this task can be accomplished by using an efficient protocol for *multi-party private set intersection cardinality (MPSI-CA)*, like those proposed in [79,80], or an approximate technique, to achieve better efficiency, like that proposed in [81] (in this case, an extra value of the privacy level k should be set to guarantee that the anonymity threshold $\frac{1}{k}$ is fulfilled). MPSI-CA is a secure multi-party-computation (SMPC) protocol, allowing any party among a group of participants, each owning a private set of items, to compute the cardinality of the intersection of these sets, without learning anything about the sets of the other participants. This way, in our case, A knows nothing about users of B (not registered to A) and vice versa.

To generalize the above case (and then, also to understand why we need to compute private set intersection cardinality among multiple parties — more than two), we introduce the following notions.

Definition 6.1. Given a cell $z \in Z_A$, we denote by L_z the set of LTSs of level 0 supporting the service s in z. Given a set of LTSs of level 0 $X \subseteq L_z$, we define now the following recursive notion:

$$\begin{aligned} c_{\emptyset} &= \left|\bigcap_{Y \in L_z} P_j^Y[z]\right| \\ c_X &= \left|\bigcap_{Y \in L_z \setminus X} P_j^Y[z]\right| - \sum_{Y \subset X} c_Y \end{aligned}$$

From Definition 6.1, it is not clear the meaning of c_X. Thanks to the following theorem, we prove that c_X is the number of users registered with all and only the LTSs of level 0 in L_z but not in X.

Theorem 6.1. *Given a cell $z \in Z_A$, and a set of LTSs of level 0 $X \subseteq L_z$, c_X is the cardinality of the set of the users who are registered with all and only the LTSs of level 0 belonging to the set $L_z \setminus X$.*

Proof. We proceed by induction on the cardinality of the set X, by proving that the property stated in the theorem holds when $|X| \leq k$, for any $k \geq 0$.

Basis. (i.e., $|X| \leq 0$). In this case, $X = \emptyset$, and then, trivially, $c_\emptyset = \left|\bigcap_{Y \in L_z} P_j^Y[z]\right|$ represents the set of the users who are registered with all and only the LTSs of level 0 belonging to the set L_z.

Induction. We have to show that if the theorem statement holds for any set X with $|X| \leq k$, it also holds for any set X with $|X| \leq k+1$. According to Definition 6.1, $c_X = \left|\bigcap_{Y \in L_z \setminus X} P_j^Y[z]\right| - \sum_{Y \subset X} c_Y$.

The term $\left|\bigcap_{Y \in L_z \setminus X} P_j^Y[z]\right|$ counts all the users who are registered with *all* the LTSs of level 0 belonging to the set $L_z \setminus X$. We denote by A, this term.

However, A also includes those users who are registered also with LTSs in X. To prove induction, we have to show that c_X does not also count these users. This actually happens thanks to the subtractive second term. Indeed, by the inductive hypothesis, for any $Y \subset X$ (and, thus, $|Y| \leq k$), c_Y represents the set of the users who are registered with all and only the LTSs of level 0 belonging to the set $L_z \setminus Y$. Then, $\sum_{Y \subset X} c_Y$ represents the number of users who are registered with all the LTSs of level 0 belonging to the set $L_z \setminus X$ and *at least one* LTS of level 0 in X. We denote by B, the term $\sum_{Y \subset X} c_Y$.

As A is the number of users who are registered with all the LTSs of level 0 belonging to the set $L_z \setminus X$ and B is the number of users who are registered with all the LTSs of level 0 belonging to the set $L_z \setminus X$ and at least one LTS of level 0 in X, we have that the difference $A - B = c_X$ represents the number of users who are registered with all the LTSs of level 0 belonging to the set L_z who are not registered with any LTSs of level 0 in X. The proof is then concluded. □

With this result in hand, we can now describe how each LTS of level 0 in L_z computes the number of users belonging to cell z to notify it to the higher LTS level. This number, denoted by n_z, is independent of the LTS counting it because it counts the overall number of users not denying the service s registered with any of the LTSs in L_z.

We assume that, preliminarily, in each 0-zone A_j, all the LTSs of level 0 operating on it (L_z, for each z in A_j) know each other (this task can be supported by ST). The protocol can proceed as follows, cell by cell (in parallel) in the 0-zone:

- For each subset of LTSs in L_z, the MPSI-CA (multi-party private set intersection cardinality) protocol is executed among the corresponding sets of registered users to obtain the cardinality of the intersection of such sets. The overall effort of this step is $2^{|L_z|} - |L_z| - 1$ MPSI-CA executions. This is not prohibitive because we expect, in real-life situations, very few LTSs per 0-zone. Moreover, due to the parallel execution of the intersections, the overall computation is bounded by one MPSI-CA execution among $|L_z|$ sets.
- Each LTS in $L \in L_z$ notifies to all the LTSs of level 1 in $up_0(L)$ the following information: (1) $|P_j^L[z]|$, and (2) the result of any executed intersection in which L is involved. To avoid duplication of communication, we could establish some P2P protocol among the involved LTSs. For simplicity, we omit this aspect. Eventually, each LTS of level 1 is aware of all the needed information to compute c_Xs, for each subset $X \subseteq L_z$.
- At this point each LTS of level 1 can compute n_z. This is done by choosing any LTS of level 0, say A, and then by computing: $n_z = |P_j^A[z]| + \sum_{\{Y \in 2^{L_z} \mid A \in Y\}} c_Y$. This immediately follows from Definition 6.1 and Theorem 6.1, due to the fact that, for two distinct subsets of LTSs X_1 and X_2, the corresponding c_Xs refer to disjoint sets. Therefore, the second term of the previous expression counts all the users not registered with A.
- The LTSs of level 1 forward the value n_z to the LTSs of level 2, and so on to the root.
  The value n_z is forwarded by the LTSs of level 1 to the higher level, and so on.

In the next example, we give an instance of the application of our protocol to a case with three LTSs of level 0 and 1 LTS of level 1.

Example 6.1. Suppose that, for a given cell z in the 0-zone A_j, there are three LTSs of level 0 operating on it, i.e., $L_z = \{A, B, C\}$. Suppose that $up_0(A) = up_0(B) = up_0(C) = \{D\}$, that is the three LTSs of level 0 are down just 1 LTS of level 1. Now, we show how our protocol is executed. For simplicity, we denote by $I_X = P_j^X[z]$, for any LTS $X \in L_z$. The overall number of MPSI-CAs to perform is 4 (i.e., $2^3 - 3 - 1$). They are: (1) $I_{A,B,C} = |I_A \cap I_B \cap I_C|$; (2) $I_{A,B} = |I_A \cap I_B|$, (3) $I_{A,C} = |I_A \cap I_C|$, and (4) $I_{B,C} = |I_B \cap I_C|$. Once they are executed, $I_{A,B,C}$, $I_{A,B}$, $I_{A,C}$, $I_{B,C}$, $|I_A|$, $|I_B|$, and $|I_C|$ are notified to D.

At this point, D is able to compute the following:

$$\begin{array}{lcl}
c_\emptyset & = & I_{A,B,C} \\
c_{\{A\}} & = & I_{B,C} - \sum_{Y\, subset\{A\}} c_Y = I_{B,C} - I_{A,B,C} \\
c_{\{B\}} & = & I_{A,C} - \sum_{Y\, subset\{B\}} c_Y = I_{A,C} - I_{A,B,C} \\
c_{\{C\}} & = & I_{A,B} - \sum_{Y\, subset\{C\}} c_Y = I_{A,B} - I_{A,B,C} \\
c_{\{A,B\}} & = & |I_C| - \sum_{Y\, subset\{A,B\}} c_Y = |I_C| - (c_{\{A\}} + c_{\{B\}} + c_\emptyset) = \\
 & = & |I_C| - (I_{B,C} - I_{A,B,C} + I_{A,C} - I_{A,B,C} + I_{A,B,C}) = \\
 & = & |I_C| - I_{B,C} - I_{A,C} + I_{A,B,C} \\
c_{\{B,C\}} & = & |I_A| - \sum_{Y\, subset\{B,C\}} c_Y = |I_A| - (c_{\{B\}} + c_{\{C\}} + c_\emptyset) = \\
 & = & |I_C| - (I_{A,C} - I_{A,B,C} + I_{A,B} - I_{A,B,C} + I_{A,B,C}) = \\
 & = & |I_C| - I_{A,C} - I_{A,B} + I_{A,B,C} \\
c_{\{A,C\}} & = & |I_B| - \sum_{Y\, subset\{A,C\}} c_Y = |I_B| - (c_{\{A\}} + c_{\{C\}} + c_\emptyset) = \\
 & = & |I_B| - (I_{B,C} - I_{A,B,C} + I_{A,B} - I_{A,B,C} + I_{A,B,C}) = \\
 & = & |I_B| - I_{B,C} - I_{A,B} + I_{A,B,C}
\end{array}$$

Now, D can compute n_z, by choosing any LTS, say A:

$$n_z = |I_A| + \sum_{\{Y \in 2^{L^z} \mid A \in Y\}} c_Y = |I_A| + (c_{\{A\}} + c_{\{A,B\}} + c_{\{A,C\}})$$

6.3. Computational complexity

In this section, we evaluate the computational complexity of our method by considering the most general case of multiple services managed by possible overlapping LTSs.

We denote by N_u the total number of users in the system and by N_s the total number of services offered in the system.

We have to analyze the computation complexity of the three functions supported by our method, namely registration, position notification, and LBS-request processing.

6.3.1. Registration

Each LTS L manages a given number of users $N_u(L)$ located in the zones it manages. Moreover, L manages a number $N_s(L)$ of services. Since each user may adhere to different services, a way in which this mapping can be implemented is through a hash table that associates each service with the list of users adhering to such service.

In terms of space, the size of the hash table is $N_u(L) \cdot N_s(L)$.

Since, $N_u(L) \in O(N_u)$ and $N_s(L) \in O(N_s)$, the space required for each LTS is $O(N_u \cdot N_s)$.

We observe that this is the same size required for a centralized approach with a single LTS managing all the users and all the services simultaneously.

Each time a user joins the system, the user will be inserted in each list associated with each service to which the user adheres. Since each insertion can be performed in constant time ($O(1)$), the total cost of the registration phase for the LTSs of level 0 is $O(N_s)$.

6.3.2. Position notification

Concerning the phase of position notification, we distinguish between the LTSs of level 0 and LTSs of higher levels.

Each LTS of level 0 receives the actual positions of the users and updates the current positions. The cost of this operation is $O(N_u)$ in the worst case in which all the users of the system are located in the same 0-zone.

In addition, as discussed in Section 6.2, each LTS L of level 0 computes the aggregation mapping through a number of MPSI-CA executions. We denote by $f_m(N_L, N_c)$ the cost of a single MPSI-CA execution among N_L parties and sets of cardinality N_c. In our case, N_L is the number of LTSs of level 0 covering the same 0-zone of L and sharing with L at least one service. N_c represents the number of users in a cell.

We recall that the number of MPSI-CA executions is $2^{N_L} - |N_L| - 1$. Then, we can conclude that the total cost of the position notification performed by L is $O(N_s \cdot (2^{N_L} \cdot f_m(N_L, N_c) + N_u))$.

To understand if the above cost leads to infeasibility for large real-life inputs, we have to analyze the two components 2^{N_L} and $f_m(N_L, N_c)$. From the other components, we do not have sources of potential infeasibility (note that the cost depends linearly on the number of services and the number of users).

Let us start from 2^{N_L}. We observe that, despite its exponentiality, this component does not affect the actual scalability of the method. Recall, indeed, LTSs of level 0 are organizations providing a local service in a 0-zone. Even in a very dense scenario, we expect that the number of different organizations covering the same 0-zone is very small (a significant growth of overlapping LTSs in the same 0-zone is not plausible from the business point of view).

Concerning $f_m(N_L, N_c)$, we observe that there are highly efficient private set intersection techniques. For instance, [80] proposes a technique with a cost that grows linearly in the number of participants (N_L in our case) and the maximum size of the sets (N_c in our case).

Concerning higher-level LTSs, they will receive the aggregation mapping from lower-level LTSs. Since the aggregation mapping counts (in aggregate form) all the users in the various cells, it requires a space of $O(N_s \cdot N_u)$ to be stored and a time $O(N_s \cdot N_u)$ to be updated.

Therefore, from the analysis above, it appears that our method is feasible and scalable in realistic settings.

6.3.3. LBS-request processing

In this phase, the user submits an LBS request for a given service. Therefore, the cost of this phase is independent of the number of services. The computational cost of this phase derives from the multiple invocations of Algorithm 1.

We evaluate a single invocation of $\mathcal{A}_{cloak}$. This involves two operations: first, the generation schema $\mathcal{R}$ is invoked, followed by the invocation of the Cloaking Oracle $\mathcal{O}_A$.

We recall that $\mathcal{O}_A$ may fail or return a cloaking area. So the computational cost may change. We denote by $f_c(N_u)$ the cost of the invocation of $\mathcal{O}_A$ in the worst case when invoked with N_u users as input. We denote by $f_r(N_u)$ the cost of the invocation of $\mathcal{R}$ when invoked on the aggregation mapping involving N_u users.

We can assume that the $f_r(N_u) < f_c(N_u)$. Indeed, $\mathcal{R}$ simply computes a fixed re-distribution of the users (for example the uniform distribution) in the cells in the area A.

On the other hand, $\mathcal{O}_A$ is expected to compute more complex operations on the users in the area A. Then, the cost of the single invocation of $\mathcal{A}_{cloak}$ is less than $2 \cdot f_c(N_u)$.

Now, $\mathcal{A}_{cloak}$ may be invoked several times according to the fact that $\mathcal{O}_A$ fails or not. We denote by l the number of invocations. Then, we can conclude that the total cost computational cost of our hierarchical approach is less than $2 \cdot l \cdot f_c(N_u)$.

Therefore, the cost of this operation is $O(l \cdot f_c(N_u))$.

We highlight the following points:

1. This is a worst-case assumption. Indeed, the algorithm $\mathcal{A}_{cloak}$ is invoked several times when the underlying oracle $\mathcal{O}_A$ fails. However, we can expect, for real algorithms, that the cost of $\mathcal{O}_A$ when fails is smaller than the cost of $\mathcal{O}_A$ when returns a cloaking area.
2. As shown in Section 9.5.3, for a specific Cloaking Oracle, the computational time of the Cloaking Oracle (and then of our approach) is negligible compared to realistic transmission time.

6.4. Implementation aspects

As highlighted earlier, our hierarchical distributed LTS system enables the possibility to adopt the emergent edge–cloud architecture [82,83] for mobile applications, like that presented in [84]. Indeed, this paper is a state-of-the-art architecture that deploys cloud servers at the network edge and designs the edge–cloud as a tree hierarchy of geo-distributed servers. Therefore, this architecture perfectly fits our case. Indeed, even though the use of cloud resources appears necessary for any LTS-based solution, to serve the spatial peak loads from mobile users, our LTS organization enables a hierarchical architecture of edge cloud, which allows incremental local management of the peak loads across different tiers of cloud servers to better distribute the mobile workloads [85] and reduce network latency. Differently from the approach presented in [84], in which a workload placement algorithm is defined to hierarchically distribute the computation load, we expect that the decentralization of computation is a *for-free* side-effect of the inherent locality of cloaking areas for even high privacy levels. Indeed, supposing we consider a 5G scenario [86], if we implement the LTSs of level 0 at the *small cells*, and then, the higher levels at the higher tiers (by using the various levels of *macro cells*), placing the highest LTSs at the (traditional) cloud, we expect that the most queries can be resolved by LTSs of low level (0 or 1), with a significant positive impact both on the overall computational workload and, importantly, on service latency. However, a challenge could be to include in our system a certain degree of flexibility in the hierarchy, leveraging virtualization techniques and optimization techniques. The idea could be to achieve, in this way, zones whose size and shape change over time, depending on service demand.

Even though, in this paper, we do not provide an edge–cloud implementation of our system, the aim of this section is to highlight this implementation aspect as a relevant nice feature of our approach. In Section 9, we analyze the impact on the latency, throughput, and traffic overhead of a 4-tiers edge–cloud implementation against a traditional 1-tier implementation by simulating users' requests in different configurations, obtaining confirmation of our expectations.

6.5. Case study

To give a clearer intuition of the entire process put in place by our system, we describe in this section how this process works in a real-life case. We refer again to Fig. 2. Therein, we have three adjacent 0-zones, colored in green, red, and blue, respectively, managed by three LTSs of level 0, say L_g, L_r, and L_b, respectively.

We suppose that the three LTSs are implemented on three base stations providing the users with the mobile connectivity in these 0-zones.

The union of the three 0-zones forms a 1-zone managed by the black LTS of level 1, say L_{bk}. This LTS is deployed on a (remote) cloud server accessible from the three LTSs of level 0.

We suppose two location-based services are provided, say s_1 and s_2.

s_1 refers to a medical service, in which a user requires the available cardiology facilities in proximity of their position.

s_2 is a gambling service, in which a user requires the available gambling houses in proximity of their position. Clearly, although for different reasons, both services require a given degree of privacy.

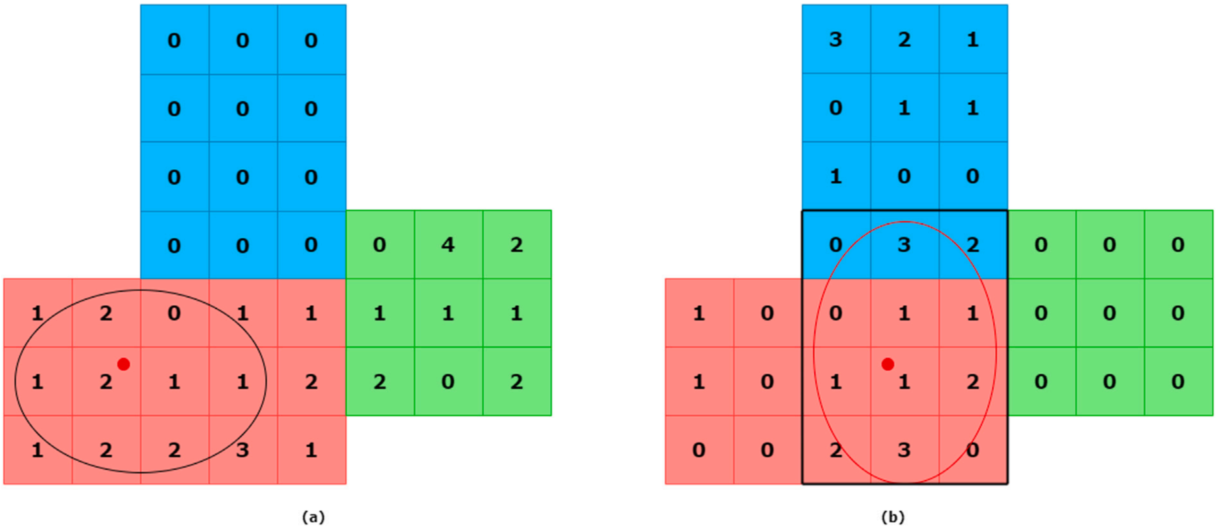


Fig. 5. Aggregation mapping for s_1 (Figure (a)) and s_2 (Figure (b)).

In our case study, L_g manages the service s_1, L_b manages the service s_2, L_r and L_{bl} manage both the service s_1 and s_2.

According to the figure, L_g manages 13 users, L_r manages 21 users, and L_b manages 14 users.

However, for ethical reasons, among the 21 users managed by L_r, 10 users do not provide the consensus to the use of their position for the service s_2. Therefore, the view of L_{bl} in terms of aggregation mapping is different from Fig. 2.(b).

Specifically, L_{bl} has two views (one for each service) represented in Fig. 5.

We observe that, since the 13 users managed by L_b are interested only in s_2, the aggregation mapping for s_1 (Fig. 5.(a)) counts 0 users for the users in the blue 0-zone.

Similarly, the aggregation mapping for s_2 (Fig. 5.(b)) counts 0 users for the users in the green 0-zone.

Now, consider a user u in the red 0-zone asking for the service s_1 with a privacy requirement $k = 14$. u submits the request to L_r which invokes the Cloaking Oracle $\mathcal{O}_{\mathcal{A}}$. Since a sufficient number of users is present in the red 0-zone, $\mathcal{O}_{\mathcal{A}}$ returns a cloaking area (depicted with a black border in Fig. 5.(a)). Then, L_r submits the LBS request to the service provider of s_1 (with the cloaking area in place of the actual position of u). The response is multicasted to all the users in the cloaking area.

Consider now another user u' in the red 0-zone asking for the service s_2 with the same privacy requirement $k = 14$. u submits the request to L_r which invokes $\mathcal{O}_{\mathcal{A}}$. However, differently from the previous case, the red 0-zone does not contain a sufficient number of users to build a cloaking area for the service s_2. Then, $\mathcal{O}_{\mathcal{A}}$ fails, and L_r forwards the request to L_{bl}. L_{bl} invokes $\mathcal{A}_{cloak}$. In turn, $\mathcal{A}_{cloak}$ invokes $\mathcal{O}_{\mathcal{A}}$ (after computing a redistribution of the users).

$\mathcal{O}_{\mathcal{A}}$ returns a cloaking area (depicted with a red border in Fig. 5.(b)) including users of the red and blue 0-zones (adhering to service s_2). Finally, $\mathcal{A}_{cloak}$ returns the approximate cloaking area (depicted with a black border in Fig. 5.(b)). L_{bl} submits the LBS request to the service provider of s_2. L_{bl} forwards the LBS response to L_r and L_b which, in turn, forward it to all the users in the cells forming the approximate cloaking area.

7. A possible business model

An interesting aspect to consider is the plausibility of the proposal from a market perspective. In this section, we outline a potential business model to demonstrate how the involved parties – those managing the location-based services, the location-trusted service, and the communication infrastructure – are organized to establish an ecosystem supporting the business. It is worth noting that we do not have the ambition, in this paper, to develop in detail a business model in a strict sense (for example, according to the *Business Model Canvas* [87]), which would be outside the scope of the presented research. However, we consider it important for the applicability of our proposal, to highlight the basic elements of a possible business model.

In this section, we refer to the notions introduced in the description of the enhanced model given in Section 6.

An important aspect, arising from the enhanced model, is that a certain organization L_j^0 can manage multiple 0-zones, and a certain 0-zone A_j can be managed by multiple LTSs of level 0. This competence overlap may occur also at the higher levels of the hierarchy.

This feature, whose impact on the theoretical framework has been discussed in Section 6 – primarily concerning position notification – enables the development of complex and flexible systems. It also allows for competition among different organizations interested in a given LTS activity.

Indeed, our distributed LTS system facilitates the presence of multiple players in the system, potentially supporting specialization in a subset of services, market competition, and, importantly, the fragmentation of the management of personal data. The latter feature, even though it is not a strong guarantee from the side of privacy threat, certainly gives benefits, also reducing the impact of a possible data breach.

We will now describe the business model we have envisioned, detailing how the *values* are *created*, how they are *delivered*, and how they are *captured* (and sold).

The *values* created in the whole system are multiple.

- TSP creates the value of *communication infrastructure and edge–cloud features* (CIE).
- SP creates the value of *location-based services* (LBS), with regional configuration, and possible service heterogeneity.
- The LTSs of level 0 create the value of *privacy support for the users* (UPS) on the selected services.
- The LTSs of level i ($i > 0$) create the value of *privacy support for the LTSs* (SPS) of level $i - 1$ on all the services required by these LTSs.
- The users collaboratively (through LTSs) create the value of *privacy tool* (UPT), to enable LTS services. They implicitly also create the value of information related to the required services. However, the aim of privacy-preserving LBS approaches is to avoid that this value can be associated with single individuals.

The above values are delivered in this way:

- CIE is delivered through a physical infrastructure and thanks to the agreement of TSP with SP and LTSs.
- LBS is delivered through a global digital platform like, for example, a social network [88]. This option seems realistic because location-based services are already provided by the major social networks, so enabling strong privacy features could enlarge both the customer base and the service segments.
- UPS are delivered through the digital platform above (in which a dedicated space is reserved for LTSs), but also through local communication features provided by ST for what concerns local multicast.
- SPS is delivered through the digital platform.
- Finally, UPT is delivered through personal smartphones (and thus the services offered by TSP) to notify their positions.

The values are captured and sold as follows:

- The 0-zones are *offered* in the market by ST with the associated services.
- They are acquired by LTSs of level 0 also partially on some services. This allows us to capture the value CIE.
- A group of adjacent LTSs of level i purchases the collaboration of an LTS of level $i+1$ to forward requests that cannot be resolved from a single LTS of level i. This allows us to capture the value SPS.
- A user u, located in a given 0-zone A_j, querying on a service $s \in S_A \setminus deny\_s(u)$, purchases the anonymization service offered by one of the LTSs of level 0, say L_x^0, chosen among those belonging to the set $zero\_lts(A_j)$, such that $s \in services(L_x^0, A_j)$. The user may be charged depending on the privacy requirement (i.e., k, and the overhead imposed on the system by this requirement). This allows the user to capture the values UPS and LBS and the LTS to capture the value UPT.
  We observe that the user can query only about services that they have not marked as denied. Even though, the function $deny\_s$ regards only the construction of cloaking areas, we decide to apply this sort of *symmetry* between required and supported services, as a logical coherence of the user's activity.

8. Security analysis

In this section, we analyze the security of our solution.

Before defining our threat model, we recall the actor involved in our solution.

Actors. They are:

- **Users:** who submit LBS queries.
- **LTSs:** which:
  - process the queries of the users,
  - build the cloaking areas,
  - contact ST (see below), and
  - provide the answers to the users.
- **ST:** It represents the agreement between TSP, providing the communication infrastructure, and SP, which receives the requests from LTSs and replies to them.

As illustrated in Section 6, given a query submitted by a user on a service s, the positions that can be used for this service at any level of the hierarchy must belong to 0-zones which provide service s. For this reason, for the purpose of this section, we can consider the single-service case.

Our threat model is defined in terms of: **Assumptions**, **Adversary**, and **Security properties**.

8.1. Threat model

Assumptions. The following assumptions are considered in our threat model.

A1: All the messages exchanged between users and LTSs and between LTSs themselves have the same size and are encrypted by using a secure probabilistic encryption scheme.

A2: The LTSs are trusted.

A3: The Cloaking Oracle is reciprocal, according to Definition 4.1.

Regarding Assumption **A2**, we observe that the LTSs of level 0 know the exact positions of the users but only limited to the 0-zone of their competence. This is the standard assumption of each LBS system employing LTSs. However, the LTSs of level $i > 0$ have less power than the LTSs of level 0. Indeed, they know only the number of users for each cell of the i-zone of their competence.

Adversaries. The adversary we consider is ST, with the capability of global passive adversary. This capability derives from the facts that:

- ST is able to monitor all the messages thanks to the collaboration of TSP, which controls the communication infrastructure,
- ST, thanks to SP, can maliciously exploit the received queries,
- ST, thanks to the background knowledge of SP and/or physical tracking of the users in the cells through the antennas performed by TSP, is able to know the actual position of the users.

It is worth noting that, due to Assumption **A2**, no other adversary can exist, because a user does not access any information regarding other users.

In our threat model, we aim to guarantee the following security property.

Security Property SP: It is not possible for the adversary to link an LBS request with the identity of the user performing it with probability greater than $\frac{1}{k}$ (this property coincides with the classical notion of *location k-anonymity*).

It is easy to realize that our threat model poses the protocol within very stressing conditions. The adversary we consider results from the collusion between two distinct (generally autonomous) parties, namely SP and TSP. Whereas it is somewhat standard to consider that SP is honest but curious and that it can at most rely on some background information to break users' privacy, it is not common to assume that also the TSP colludes with the adversary. In our threat model, this fact gives the adversary a very strong power, which is the power of a global passive adversary.

8.2. Security assessment

It is evident that **SP** would be compromised instantly if ST, acting as a global adversary with the capability to monitor all system traffic, could identify the query's source. However, this is not possible according to the following reasoning. Indeed, the requests are concealed within the position-notification messages, and, owing to Assumptions **A1** and **A2**, the adversary is unable to determine when a user sends a request instead of a detailed-position-notification message. Similarly, the adversary cannot discern when an LTS forwards a request to an LTS of a higher level. However, the adversary detects that a request has been originated, because it receives such a request (forwarded by a certain LTS). The adversary can identify the origin of the query by observing the response. Clearly, if the coverage of the response that the adversary can track reduces someway the cloaking area, then the adversary is able to break **SP**. It is easy to see that this is not the case. Indeed, thanks to the multicast mechanism, the response messages are directed to all the users within the constructed cloaking area.

Therefore, it remains to prove that our method preserves the property fulfilled by the Cloaking Oracle stated in Assumption **A3** (i.e., k-reciprocity). This guarantees Property **SP**.

To do this, recall that the cloaking area is the result of either (i) the invocation of the Cloaking Oracle $\mathcal{O}_A$ (if the area can be constructed by an LTS of level 0) or (ii) the invocation of Algorithm $\mathcal{A}_{cloak}$. In case (i), due to Assumption **A3**, the above statement is clearly satisfied. In case (ii), the cloaking area is built by an LTS of level greater than 0. Recall that, in this case, Algorithm $\mathcal{A}_{cloak}$ returns an approximate cloaking area $\overline{C}_{P'}$ of $C_{P'} = \mathcal{O}_A(k, P', p')$ where $P' = \mathcal{R}(P)$, and $p' = \mathcal{S}(p) \in P'$ where $p \in P$ is the actual position of the user performing the request.

The next theorem states that our approximate construction-cloaking-area technique is reciprocal.

Theorem 8.1. *Given the geographical area A, the partition Z_A of A, the privacy parameter k, the set of position P, the generation schema $\mathcal{R}$, the one-to-one mapping $\mathcal{S}$, and the approximate cloaking area algorithm $\mathcal{A}_{cloak}$ with input k, D_P, and a position $p \in \mathcal{R}(D_P)$, then $\mathcal{A}_{cloak}$ is reciprocal.*

Proof. We have to prove that there exist at least k positions $p_1, \dots, p_k \in \overline{C}_{P'} \cap P$ such that $\overline{C}_{P'} = \mathcal{A}_{cloak}(k, D_P, x)$, for each $x = \mathcal{S}(p_i)$, for any $1 \le i \le k$.

By Assumption **A3**, being $C_{P'} = \mathcal{O}_A(k, P', p')$, there exist at least k positions $p'_1, \dots, p'_k \in C_{P'} \cap P'$ such that $C_{P'} = \mathcal{O}_A(k, P', p'_i)$, for any $1 \le i \le k$. Since $p'_i \in P'$ (for any $1 \le i \le k$), by definition of $\mathcal{S}$ (that is a one-to-one mapping) there exist at least k positions $p_1, \dots, p_k \in P$ such that $p'_i = \mathcal{S}(p_i)$ for each $1 \le i \le k$. Since $C_{P'} = \mathcal{O}_A(k, P', p'_i)$, for any $1 \le i \le k$, by definition of $\mathcal{A}_{cloak}$, $\overline{C}_{P'} = \mathcal{A}_{cloak}(k, D_P, x)$, for each $x = p'_i = \mathcal{S}(p_i)$, for any $1 \le i \le k$. Indeed, $\overline{C}_{P'}$ depends only on $C_{P'}$. Therefore, it suffices to prove that $P_r = \{p_1, \dots, p_k\} \subseteq \overline{C}_{P'}$. Let q be a position in P_r. We have to prove that $q \in \overline{C}_{P'}$. By definition of $\mathcal{S}$, there exists $q' \in P'$ such that $q' = \mathcal{S}(q)$ and $q, q' \in z$, for some $z \in Z_A$. Therefore, we show that $z \subseteq \overline{C}_{P'}$. By definition of $C_{P'} = \mathcal{O}_A(k, P', q')$, $q' \in C_{P'}$, and then $z \cap C_{P'} \neq \emptyset$. Therefore by definition of $S_{C_{P'}}$, $z \in S_{C_{P'}}$. Finally, by definition of $\overline{C}_{P'} = \bigcup_{y \in S_{C_{P'}}} y$, therefore $z \subseteq \overline{C}_{P'}$. The proof is then concluded. □

9. Experiments

This section aims to provide an experimental validation of our proposal.

9.1. Selection of a cloaking-area construction algorithm

We start by recalling that our approach is parametric with respect to any cloaking-area-construction algorithm. In the theoretical framework, we called Cloaking Oracle the underlying cloaking-area-construction algorithm. For the experiments, we consider a state-of-the-art approach, called Casper [29] described in Section 3.2.

We want to highlight that Casper is not used for a comparative analysis. Indeed, we do not propose a new cloaking-area-construction algorithm (as Casper does), but a method to apply any existing cloaking-area-construction algorithm to aggregate location data. On the other hand, Casper does not deal with the problem faced by our approach, i.e., the implementation of a distributed hierarchical LTS system.

The choice of Casper is due to its relevance in the scientific literature (as both the publication venue and the very high-impact witness).

In our experiments, we use the JAVA implementation of Casper provided in [89]. We modified this JAVA project to implement our solution.

9.2. Experimental environment

Due to the computational effort required to conduct the experiments, to reduce the time to complete them, we employed three personal computers in parallel equipped with different specifications: 1.8 GHz Intel i7-8850 CPU and 16 GB of RAM, 2.5 GHz Intel i7-6500 CPU and 12 GB of RAM, and 1 GHz Intel i5-1035G1 CPU and 8 GB of RAM respectively. In particular, we evaluated the performance in different scenarios with different conditions (basically, each scenario corresponds to a plot). Since the scenarios can be tested independently, each computer is employed to obtain the results of a subgroup of scenarios.

As in [90,91], in our experiment, the user data are generated by using the Thomas Brinkhoff data generator [78]. Furthermore, we also consider the mobility pattern, provided by the generator, to study the performance of our approach when users move into the map.

We chose to conduct our experiments on a portion of the center of the city of Reggio Calabria (Italy) and analyze the performance as the number of users varies. In Fig. 6, the considered 2 km × 2 km map provided by OpenStreetMap (OSM) [92] is depicted, including the distribution of 6000 users generated by [78] at a given instant.

Note that, in principle, the results we obtain may depend on the real-life region of analysis. However, it is important to consider that the selection of a real-life region, as opposed to a synthetic one, is primarily aimed at establishing a connection with the real world. Nonetheless, this choice would not have been sufficient if only one (or a few, even real) people distributions were considered. Therefore, to simulate different, also very distant from each other, conditions, and then, virtually, different regions with different habits, we generated many set-ups for the experiments. It is easy to realize that the actual factor that can impact our experiments is the density of users' distribution. Across different set-ups, we varied this value significantly, ranging from sparse to very dense distributions. On the other hand, the local topology of cities (referring to the size we considered in the experiments) is somewhat similar. We chose a medium-small (Italian) city to have a higher maximum density than a large city because streets are more narrow and buildings are smaller. This allowed us to stress the experiments with a wider range of densities. On the basis of the above reasoning, we avoided repeating the experiments not only for different densities but also for different cities, also to make the effort more feasible.

As for the implementation of our approach, in favor of the clarity of the analysis, we referred to the basic tree-like LTS hierarchy, with a single-service assumption and square zones. However, we argue that the significance of our experimental analysis is not affected. Indeed, the presence of multiple services has minimal impact on the performance we evaluated, i.e., effectiveness and network performance. Concerning effectiveness, the presence of multiple services has no impact at all (since each cloaking area is built for a service regardless of the presence of other services). Instead, regarding network performance, it is easy to see that the computational time introduced by the application of cryptographic algorithms (as those discussed in Section 6) is negligible compared to the network latency which becomes the main factor to consider when analyzing the network performance of our proposal.

We considered a four-layer LTS hierarchy.

Specifically, there is a single 3-zone of size 2 km×2 km. This 3-zone is divided into four 2-zones of size 1 km × 1 km. Each 2-zone includes twenty-five 1-zones of size 200 m×200 m. In turn, each 1-zones includes twenty-five 0-zones of size 40 m×40 *m*. Finally, we consider cells of size 8 m × 8 m. Therefore, the 0-zones include 25 cells, the 1-zones include 625 cells, the 2-zones include 15,625, and the 3-zone includes 62,500 cells.

We observe that such configuration of zones and LTSs is aligned to the edge–cloud implementation discussed in Section 6.4. Indeed, the 0-zones are of the order of 5G small cells.

Concerning the implementation of Casper, we considered the same conditions (same geographical area, number of users, positions of the users, moving patterns, and network conditions). Obviously, since Casper is centralized, no LTS hierarchy is present, and then the queries are processed by a single LTS coinciding with our LTS of level 3.

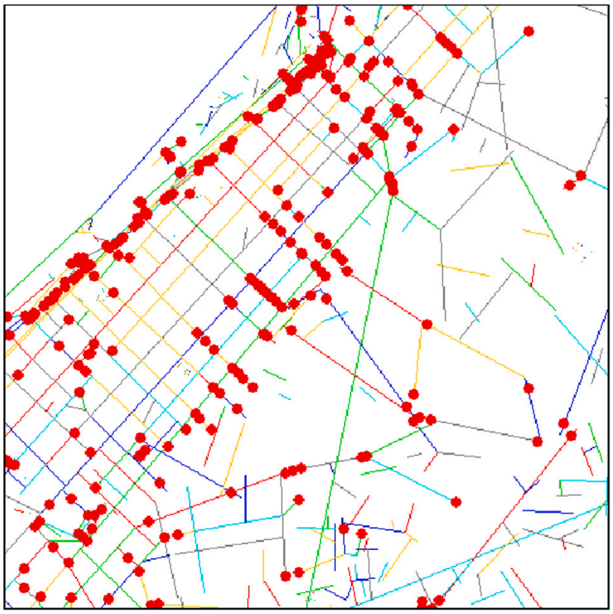


Fig. 6. Distribution of the users in the selected area of Reggio Calabria.

9.3. Statistical significance of data

In our experiments, we used the Brinkhoff data generator [78] to simulate the positions of users in the considered geographic area.

[78] is a state-of-the-art reference as evidenced by the high number of citations and its use even in very recent top-quality papers [91,93].

In this section, we evaluate whether the use of this generator is suitable for our simulations.

The parameter we are interested in is the cloaking area size. It is used to evaluate the effectiveness [10] in a static scenario (see Section 9.5.1) and dynamic scenario (see Section 9.5.2). In both scenarios, we performed several measurements of the cloaking area size and averaged them. We evaluated the statistical significance of these measurements through the Student's t-test [94].

Specifically, we assume the null hypothesis the following H_0 =“The samples obtained for Casper and those obtained for our approach have the same average values”. Indeed, we want to show that no (significant) difference exists between our approach and Casper in terms of cloaking area size. We compute the p-value for each pair of samples of Casper and our approach used in our experiment and accept the null hypothesis for $p > 0.05$.

Specifically, concerning the experiments of Section 9.5.1, we computed the p-value for each considered k and N in which 50 samples are collected for Casper and our approach. Concerning the experiments of Section 9.5.2, we computed the p-value for each time instant and k in which 80 samples are considered for Casper and our approach.

In all experiments, the p-value resulted in greater than 0.05, confirming the significance of the measurements obtained.

9.4. Goals and metrics

The objective of this experimental evaluation is twofold:

1. We want to study how much working on aggregate data with our method may degrade the quality of the cloaking-area construction algorithm result
2. We want to highlight the benefits deriving from the decentralized architecture in terms of network performance when an edge–cloud implementation is adopted.

Concerning (1), we consider the *cloaking-area size* as a metric. Once a privacy level k is fixed, the objective is to minimize the cloaking area, ensuring it includes at least k users. In the literature, this property is referred to as *effectiveness* [10]. As discussed in Section 1, large cloaking areas incur high processing overhead from the LBS side and network costs, attributed to the high number of candidate results to return to the LTS. In our business model, consistent with what is stated in the literature [10], users are charged based on the required privacy level and the overhead that this requirement imposes on the system. This metric is first assessed in a static scenario by capturing the cloaking area required by a user at a fixed instant in time. Subsequently, we evaluate how the cloaking area size changes in a dynamic scenario in which users move within the territory.

Concerning (2), we consider three metrics for network performance: latency, throughput, and position notification bytes.

The latency is defined as the time needed to solve LBS queries. The throughput is related to latency and is defined as the quantity of (useful) bytes received by the users in the unity of time when they perform an LBS request. The position notification bytes represent the number of bytes received from the LTS(s) during the position notification phase.

9.5. Experiments

We discuss now the methodology followed and the results obtained for each metric in our experiment validation.

9.5.1. Cloaking area size: static scenario

These experiments aim to show that the extra size of the cloaking area returned by our method with respect to the underlying cloaking-area-construction algorithm (Casper) is very limited. Therefore, the price we pay in terms of minimality is acceptable.

Methodology. Two types of experiments are performed.

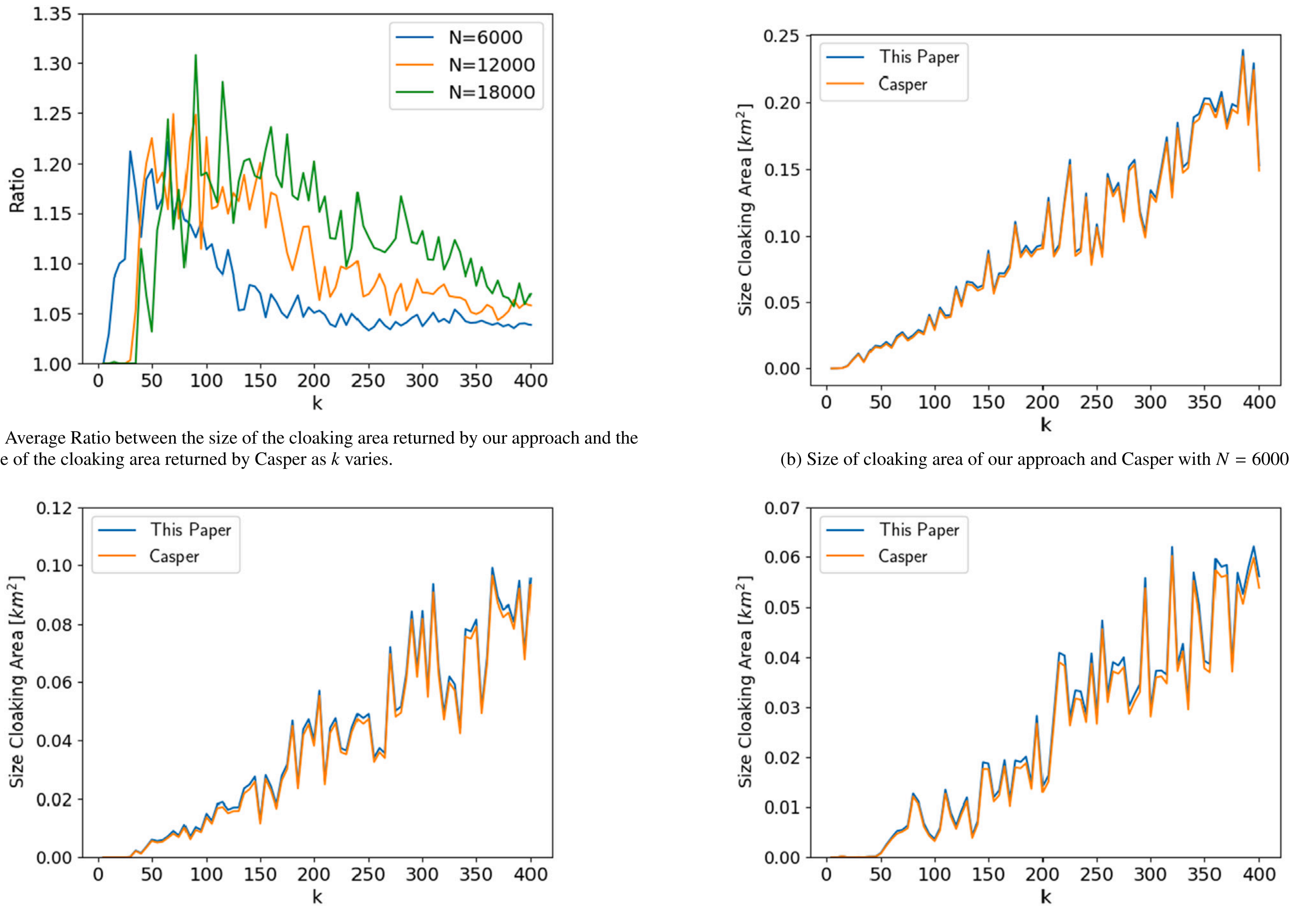
1. We fixed the total number of users in the considered area of 2 km × 2 km and measured the size of the cloaking area as the privacy requirement k varies.
2. We fixed the value of k and studied the size of the cloaking area as N varies.

In Experiment (1), as the cloaking area varies with the user performing the query, we repeated the query 50 times for each value of k with different users and calculated the average value.

Likewise, in Experiment (2), we repeated the query 50 times for each value of N with different users and calculated the average value.

Results. We begin by presenting the results of Experiment (1). In Fig. 7(a), we display three plots illustrating the average ratio between the size of the cloaking area returned by our approach and the size of the cloaking area returned by Casper as k varies. The three plots correspond to three different values of N i.e., 6000, 12 000, and 18 000. As expected, for small values of k, the ratio is almost 1. The reason is that such queries can be resolved (in most cases) by the LTSs of level 0 and, therefore, the approximate cloaking area is equal to the cloaking area returned by Casper. As the value of k increases, the queries involve also the LTS of level $i > 0$. Thus, the ratio increases too. Anyway, as k reaches a certain value, the queries are resolved (in most cases) only by the LTSs of higher level (2 or 3). In this case, the areas returned by Casper are bigger and the approximation introduced by our cells has a smaller impact. This explains the decreasing pattern.

Now we analyze the impact of the variation of the parameter N. Basically, the plots are translated. Indeed, if the density of the users increases for the same value of k, the queries can be resolved by lower LTSs. Therefore, as N increases, we require a greater value of k before the ratio rises from 1 and a greater value of k before the ratio decreases. The maximum ratio is about 1.3 and it is obtained for a limited range of values of k. For most of the values of k the ratio ranges from 1



(a) Average Ratio between the size of the cloaking area returned by our approach and the size of the cloaking area returned by Casper as k varies.

(b) Size of cloaking area of our approach and Casper with $N = 6000$.

(c) Size of cloaking area of our approach and Casper with $N = 12000$.

(d) Size of cloaking area of our approach and Casper with $N = 18000$.

Fig. 7. Size of Cloaking area as function of k.

to 1.1. Therefore, our performance regarding such a metric appears acceptable.

Now, we report the actual size of the average cloaking areas (for our approach and Casper) as k varies for three values of N i.e., 6000, 12 000, and 18 000. The results are shown in the plots in Figs. 7(b),7(c),7(d), respectively.

As expected, such size increases as k varies. Indeed, a higher privacy level implies a larger cloaking area to include more users. However, in a wide range of values for k ($k > 45$), we observe that the percentage difference between our approach and Casper in terms of cloaking area size is smaller than 10%. This witnesses that the price we pay in terms of effectiveness is very small.

As for the impact of the variation of the parameter N, we observe that, by fixing a value of k, the size of the cloaking area for both approaches decreases as N increases. This is attributed to the fact that, with more users on the map, the level of the LTSs capable of constructing the cloaking areas is lower. From these three plots, it is easy to see that no appreciable difference in the cloaking area size exists between Casper and our approach. This shows the good results in terms of effectiveness.

Consider now Experiment (2). In Fig. 8(a), we report three plots that show as the (average) ratio between the size of the cloaking area returned by our approach and the size of the cloaking area returned by Casper as N varies.

We expect that this plot is mirrored with respect to the plot in Fig. 7(a). Indeed, for small values of N, the queries are resolved by the LTSs of higher levels. This results in greater areas with a smaller approximation. Therefore, the ratio assumes small values (even if greater than 1). As N increases, the LTSs of lower levels start to get involved in the resolution of the queries and the ratio increases. When N reaches a certain threshold only the LTSs of level 0 are involved, therefore the ratio approaches 1. This is evident for the plot with $k = 50$. We evaluate now the impact of the variation of k. For the other plots ($k = 150, k = 300$), the above effect (obtained with $k = 50$) is less evident since we considered the maximum value of $N = 25\,000$ and a greater value of N is required. Anyway, we did not consider $N > 25\,000$ since it is not realistic in the city of Reggio Calabria inside the considered area (2 km × 2 km). For completeness, in Figs. 8(b), 8(c), 8(d) we report the actual size of the cloaking area as N varies.

Clearly, in accordance with the plots in Figs. 7(b),7(c),7(d), such size increases as k increases and decreases as N increases.

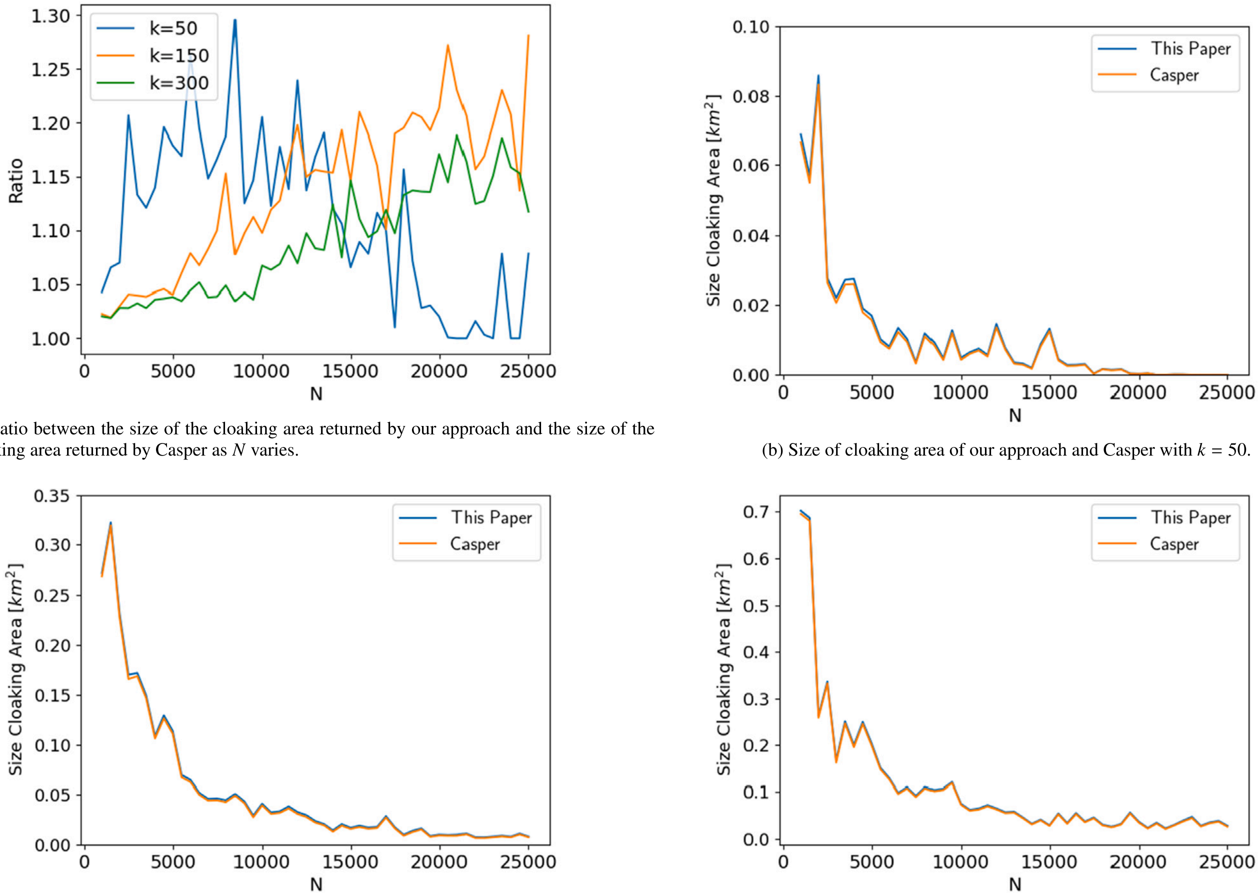
9.5.2. Cloaking area size: dynamic scenario

The goal of this experiment aligns with that of Section 9.5.1, aiming to demonstrate that the impact on cloaking area size introduced by our approach is negligible. However, unlike Section 9.5.1, this section considers a dynamic scenario where users move within the territory.

Methodology. We considered three mobility models of the users provided by [78]: *Slow*, *Middle*, and *Fast*. The Slow model corresponds to pedestrian traffic while the Fast model to vehicular traffic. The Middle model presents an intermediate behavior.

In our experiment, we measure the cloaking area size of 80 users (randomly selected) when they move on the map at different time instants. Since each user is associated with a different cloaking area instant by instant (according to their position) we selected a *representative* user and show as the size of the cloaking area obtained with our approach and with Casper, varies in the time domain.

We consider two experiments corresponding to two possible representative users.



(a) Ratio between the size of the cloaking area returned by our approach and the size of the cloaking area returned by Casper as N varies.

(b) Size of cloaking area of our approach and Casper with $k = 50$.

(c) Size of cloaking area of our approach and Casper with $k = 150$.

(d) Size of cloaking area of our approach and Casper with $k = 300$.

Fig. 8. Size of Cloaking area as function of N.

1. The first representative user is an *average* user, in which the cloaking area at each instant is given by the average of the cloaking area of the 80 users at the same instant. The limit of this approach is that the average user is not any real user and could obtain cloaking areas very different from any real user.
2. Another approach to select the representative user is to identify a real user associated with the median standard deviation σ of the distribution of ratios (over time) between the size of the cloaking area returned by our approach and the size of the area returned by Casper. This way, we select a user well representing the sample from the stability point of view because the above standard deviation is a measure of stability. Indeed, a lower value of σ, means that the user obtains similar ratios (and then, probably, similar cloaking area for our approach and Casper) as the time varies. After filtering the data by removing outlier users, we selected the *median* user as those with the median σ (if the number of users was even, we selected one of the two users with σ nearest to the median σ).

We considered both the above representative users, namely, average and median.

Results. The results of Experiment 1 are reported in Figs. 9(a), 9(b), 9(c). Specifically, therein, we report as, for $k = 50$, 150, 300, and $N = 6000$, the size of the cloaking area of our approach and Casper for the average user varies over time.

Similarly, in Figs. 10(a), 10(b), 10(c) we report as, for $k = 50$, 150, 300 and $N = 6000$, the size of the cloaking area of our approach and Casper for the median user varies over time.(Experiment 2).

In both the Experiments, the plots in the Slow scenario are less variable (and then more stable) than the plots in the Middle and Fast Scenario for both Casper and our approach.

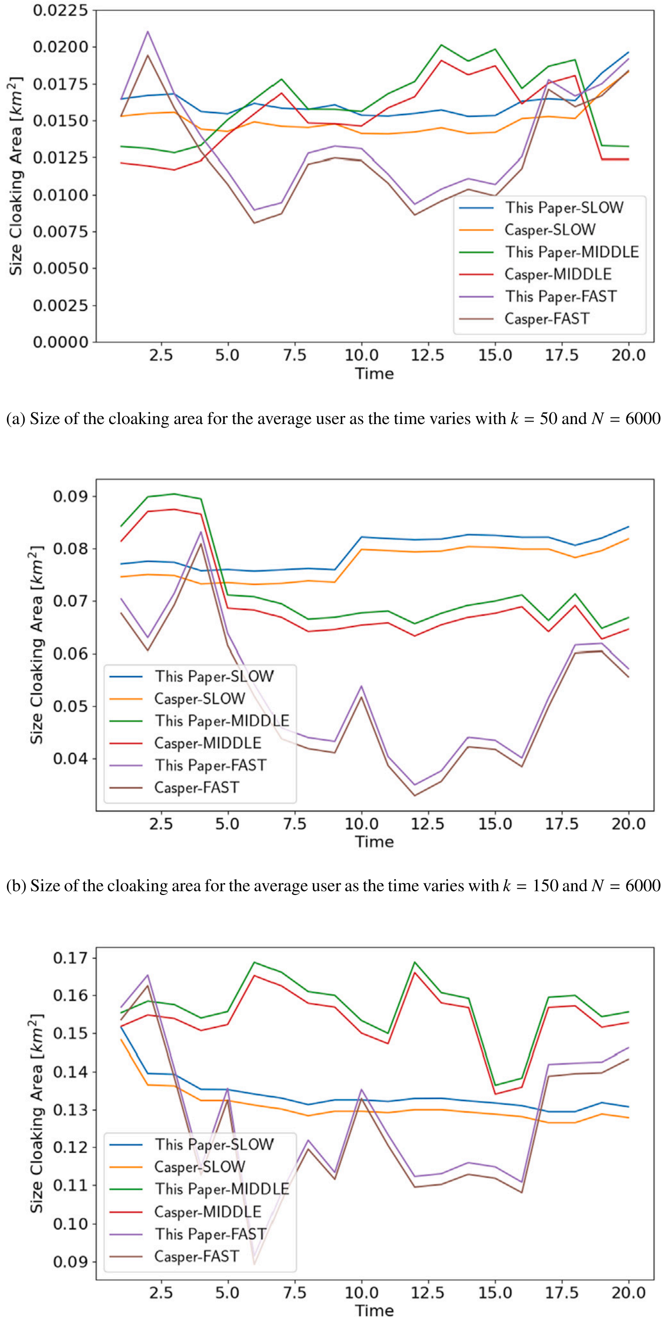
This is due to the fact that the distribution of the users does not change rapidly in the time and then the cloaking areas obtained are similar.

We discuss now the impact of the variation of the parameter k. The plots are more stable with higher values of k since they involve, in most cases, the higher LTSs, which return similar cloaking areas due to the exponential growth of the areas in Casper.

For the median user, the cloaking areas remain the same for some intervals and then suddenly change. This discontinuity arises when the user crosses different zones. This effect is smoothed for the average user.

Finally, to confirm the previous considerations, we show the average relative standard deviation of the size of the cloaking area associated with the 80 users moving into the maps. The result is shown in Figs. 11(a), 11(b), and 11(c) for different values of k and $N = 6000$.

As expected, there is no appreciable difference between our approach and Casper. Our method slightly outperforms Casper, given the lower average relative standard deviation, owing to our approximation. A minor modification in the Casper cloaking area does not result in a modification of our cloaking area. Indeed, a little modification in the Casper cloaking area does not result in a modification of our cloaking area. Clearly, the Slow scenario is more stable than the middle scenario, which is in turn more stable than the fast scenario. Finally, by evaluating the impact of the variation of k, we have that the scenarios with higher values of k are more stable than the scenarios with lower values of k. This is expected since, for higher values of k, the queries involve, in most cases, the higher LTSs, which return similar cloaking areas between Casper and our approach due to the exponential growth of the areas in Casper.

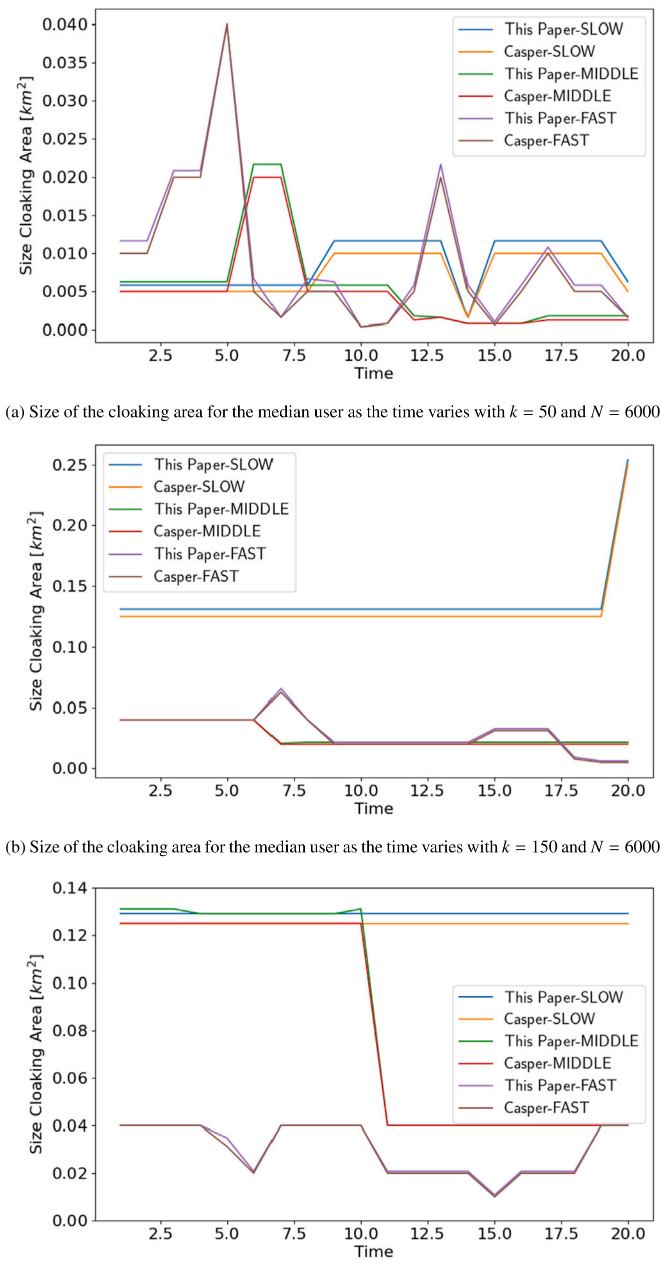


(a) Size of the cloaking area for the average user as the time varies with $k = 50$ and $N = 6000$.

(b) Size of the cloaking area for the average user as the time varies with $k = 150$ and $N = 6000$.

(c) Size of the cloaking area for the average user as the time varies with $k = 300$ and $N = 6000$.

Fig. 9. Size of the cloaking area for the average user.



(a) Size of the cloaking area for the median user as the time varies with $k = 50$ and $N = 6000$.

(b) Size of the cloaking area for the median user as the time varies with $k = 150$ and $N = 6000$.

(c) Size of the cloaking area for the median user as the time varies with $k = 300$ and $N = 6000$.

Fig. 10. Size of the cloaking area for the median user.

Overall, in both the static and dynamic scenarios, for a wide range of privacy levels ($k > 45$), the percentage difference between the cloaking area size in our approach and Casper is less than 10%. This demonstrates that the price in terms of effectiveness is limited.

9.5.3. Latency

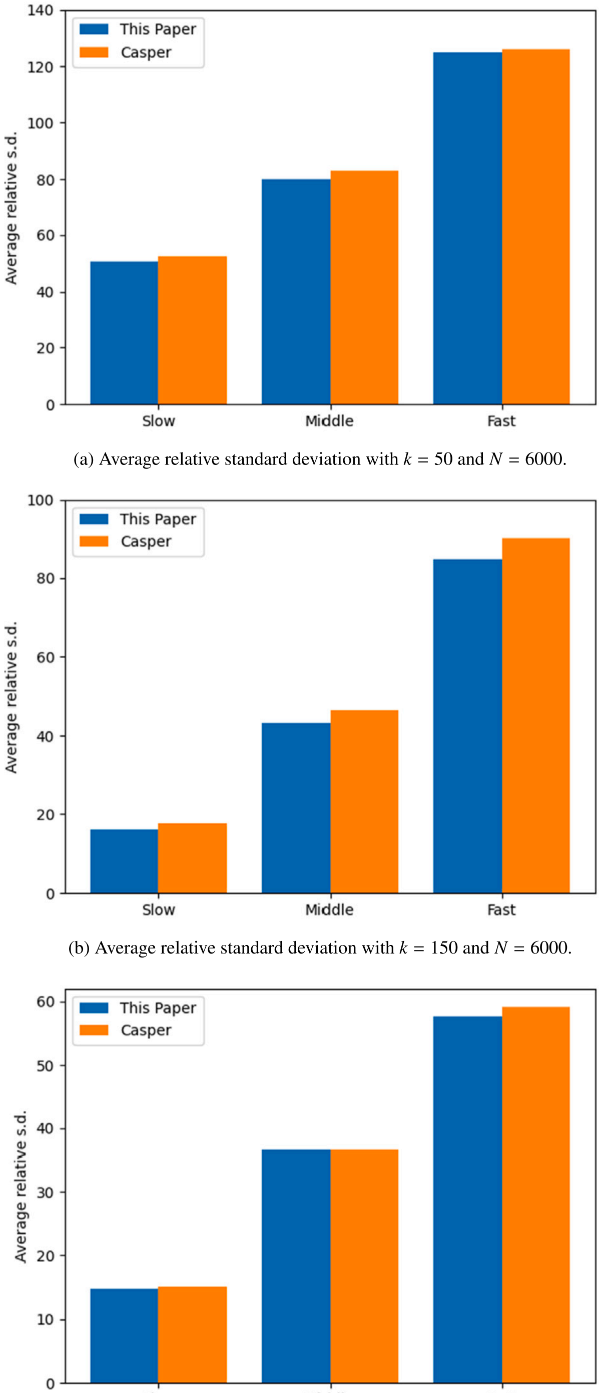
In this section, we evaluate the latency required to solve the queries. The aim of this experiment is to provide a first validation of the advantages that our decentralized solution can give when an edge–cloud implementation is adopted.

Methodology. The latency to solve the queries can be considered as the sum of two components:

1. the time to compute the cloaking area (cloaking time)
2. the network delay

Concerning the cloaking time, as reported in the original paper of Casper [29], it is negligible compared to the network delay. Indeed, it results, in the worst case, less than 0.5 ms when applied for a number of users ranging from 1000 to 50 000. According to the discussion in Section 6.3 (paragraph **LBS Request Processing**), the time required for a single execution of Algorithm 1 is less than $2f_c(N_u)$, where $f_c(N_u)$ is the cost of invocation of the cloaking Oracle (in our case $f_c(N_u) <$ 0.5 ms). Then, the time of execution of 1 is less than 1 ms. Since, in our configuration, it is invoked at maximum 3 times (see below), both in Casper and our approach, the network delay represents the predominant term.

Concerning (2), as recently reassessed in [95], in today's Internet, although there can be considerable delay variation over very short time

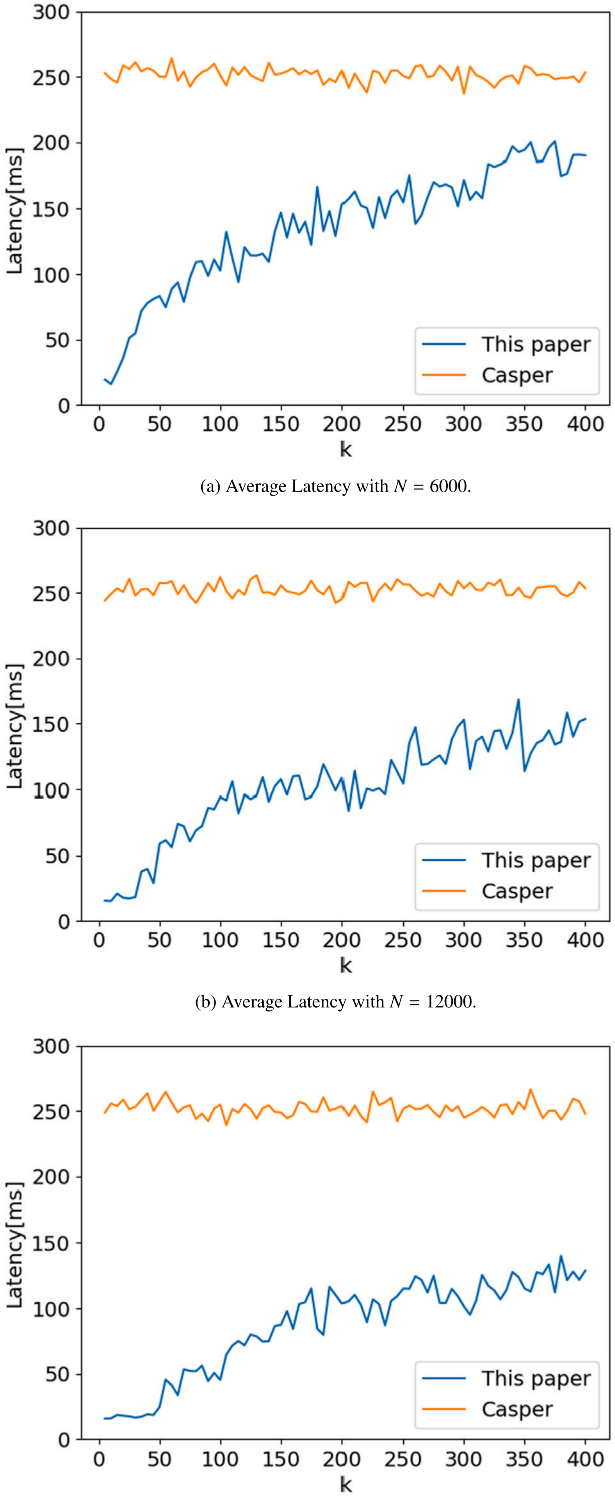


(a) Average relative standard deviation with k = 50 and N = 6000.

(b) Average relative standard deviation with k = 150 and N = 6000.

(c) Average relative standard deviation with k = 300 and N = 6000.

Fig. 11. Average relative standard deviation.



(a) Average Latency with N = 6000.

(b) Average Latency with N = 12000.

(c) Average Latency with N = 18000.

Fig. 12. Average latency.

scales, end-to-end latencies can be considered *operationally constant* on long timescales (e.g., the order of a day). Latencies can be considered operationally constant if they remain within bounds that could be considered operationally equivalent. Without the ambition to provide a conclusive evaluation of an edge–cloud implementation of our solution (not presented in this paper), but only to have a validation of this implementation direction, we study by simulation the network latency, according to the above results, by setting the bounds to 10–20 ms, 50–80 ms, 100–150 ms, and 200–300 ms for queries resolved by the LTSs of level 0, 1, 2, and 3, respectively. Indeed, we recall, we consider an edge–cloud implementation based on hierarchic tiers like that proposed in [84].

The values 50–80 ms, 100–150 ms, and 200–300 ms are obtained by following an approach similar to [96]. Specifically, we assumed that the LTSs of level 1, 2, and 3 are deployed in cloud platforms at different distances from the city of Reggio Calabria. We used the Digital Ocean platform [97] by deploying some servers in Frankfurt, New York, and San Francisco. Then, we measured the latency obtaining the above ranges (50–80, 100–150, and 200–300 for Frankfurt, New York, and San Francisco, respectively). The range 10–20 ms is instead obtained by assuming that the LTS of level 0 is in the same city (Reggio Calabria) of the users moving the map. These values of latency are measured by deploying a server at the University of Reggio Calabria.

Regarding Casper, we consider a unique LTS of level 3 of competence of the entire zone responsible for handling all the requests. In this case, the total latency is the sum of the two contributions mentioned above.

Regarding our approach, the total latency depends on which LTS resolves the query. Specifically, the query is first forwarded to an LTS of level 0. If it is able to solve it, the total latency is the same as Casper (sum of two contributions). However, since the LTS of level 0 is implemented at the edge, the network latency is much smaller. On the other hand, if the request cannot be satisfied by the LTS of level 0, the above price of latency is paid and we have to add another price of latency due to the network latency to contact the LTS of level 1 and the cloaking time. The same consideration applies if the LTS of level 1 is not able to solve the query and another price of latency is needed to contact the LTS of level 2.

We want to observe that the latency considered for our approach is slightly overestimated in favor of the fairness of the experiments. Indeed, the hierarchical edge–cloud architecture might allow lower end-to-end latency when an LTS of level $i+1$ is contacted by an LTS of level i instead of directly the user.

Results. The average latency as k varies is reported in Figs. 12(a), 12(b), and 12(c) for three values of N (i.e., 6000, 12000, 18000).

For Casper, the latency is constant (modulo random variation in the transmission delay) as k varies. Indeed, the same LTS of level 3 is involved in the query resolution.

On the other hand, regarding our approach, the latency increases with k since higher values of k involve higher-level LTSs to resolve the queries.

Concerning the impact of the variation of N, we have a dual effect. For Casper, the latency is constant since we have a single LTS of level 3 involved in the query resolution. For our approach, the latency increases as N decreases. Again, lower values of N involve higher-level LTSs to resolve the queries.

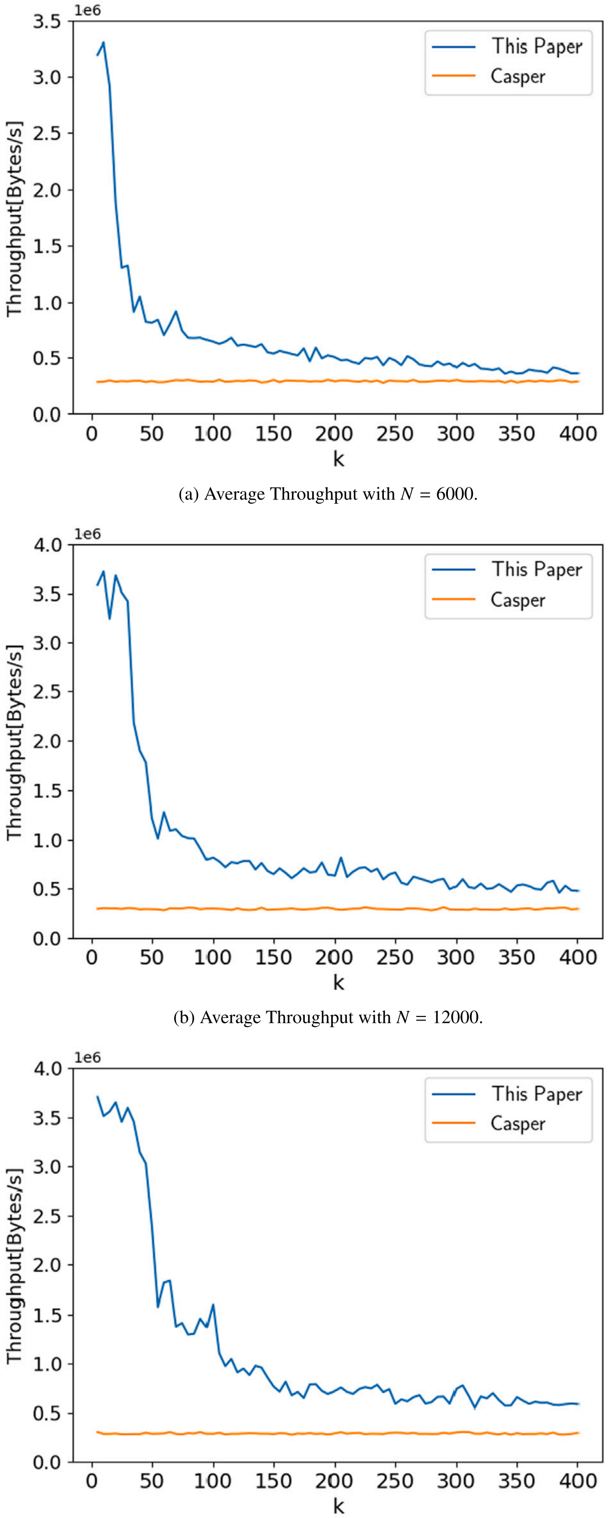
In the plots, for all the values of k considered, our approach outperforms Casper. In particular, latency is reduced from 20% to 170% according to the required privacy level. This occurs since the LTS of level 3 is never involved in building a cloaking area. Anyway, higher values of k are not required in real-life applications.

9.5.4. Throughput

Another metric we consider to evaluate the network performance is the throughput measured as the quantity of bytes contained in the query responses received by the users in the unit of time. This metric is strictly related to the network latency considered in the previous section but takes into account also the available bandwidth and the query response size.

Methodology. To perform these experiments, we considered the same parameters used in the validation of Casper [29]. Specifically, we considered a network bandwidth of 100Mbps and a query response size variable from 50–100 Kbytes (it considers a list with a variable number of candidate records of size 64 bytes). These values are also compliant with realistic real-time POIs search [98]. We also considered the same configuration of Section 9.5.3, i.e., a single LTS of level 3 for Casper and a hierarchy of LTSs (until level 3) for our approach. Again, the considered network delays are 10–20 ms, 50–80 ms, 100–150 ms, and 200–300 ms for queries resolved by the LTSs of level 0, 1, 2, and 3, respectively.

Results. The average throughput as k varies is reported in Figs. 13(a), 13(b), and 13(c) for three values of N (i.e., 6000, 12000, 18000).



(a) Average Throughput with $N = 6000$.

(b) Average Throughput with $N = 12000$.

(c) Average Throughput with $N = 18000$.

Fig. 13. Average throughput.

These plots present a mirrored behaviors with respect to plots in Figs. 12(a), 12(b), and 12(c). Indeed, for Casper, since a unique LTS of level 3 is involved, the throughput is on average constant (the variations depend on the size of the returned cloaking area). On the other hand, for our approach, the throughput decreases with k.

Concerning the impact of the variation of N, the throughput decreases with N. Again, higher values of k or lower values of N involve higher-level LTSs to resolve the queries, and then the throughput increases.

Overall, similarly to latency, the throughput improves from 20% to 170% according to the privacy level.

9.5.5. Position notification bytes

The final performance metric we consider is the number of bytes received by the LTS(s) during the position notification phase.

Methodology. We recall the configuration of LTS in Section 9.2. We have a single 3-zone divided into four 2-zones. Each 2-zone is further divided into twenty-five 1-zones, and each 1-zone is divided into twenty-five 0-zones. Each 0-zone includes 25 cells. Overall, the 3-zone includes 62,500 cells. Clearly, in a centralized approach using Casper a single LTS is employed covering the entire 3-zone.

To evaluate the exchanged bytes, we consider a number N of users in the entire 3-zone. In Casper, each user provides directly their GPS coordinates to the unique LTS of level 3. By considering a size of 11 bytes for such coordinates [99], the total number of received bytes is $11 \cdot N$ bytes for each position notification.

As for our hierarchical approach, the LTSs of level 0 receive GPS data directly from the users. In our configuration, we have $25 \cdot 25 \cdot 4 = 2500$ LTS of level 0. Then, on average, each LTS of level 0 receives $\frac{11 \cdot N}{2500}$ bytes for each position notification.

Consider now, LTSs of level higher than 0. They do not receive the coordinates from the users but just the values of the aggregation mapping i.e., the number of users for each cell. This value depends on the distribution of the user. Then, we employed the Brinkhoff data generator [78], to a distribution of N users in the 3-zone and took the maximum number of users in the cells. We repeated this experiment 50 times. We denote by x the average maximum number of users in the cell. As a simplifying worst-case assumption for our approach, we assume each cell contains x users.

The value N defines the number of bytes to transmit from an LTS of level i to an LTS of level $i + 1$. For example, if x=200, then 1 byte is needed for each cell (until 256 users). We denote by b the number of bytes needed to count x users in a single cell.

Therefore, we obtain that:

1. Since each LTS of level 1 manages 625 cells, it receives $625 \cdot b$ bytes from the LTSs of level 0
2. Since each LTS of level 2 manages 15625 cells, it receives $15625 \cdot b$ bytes from the LTSs of level 1
3. Since the LTS of level 3 manages 62500 cells, it receives $62500 \cdot b$ bytes from the LTSs of level 2

Results. We performed our simulation for three values N i.e., 6000, 18 000, 25 000.

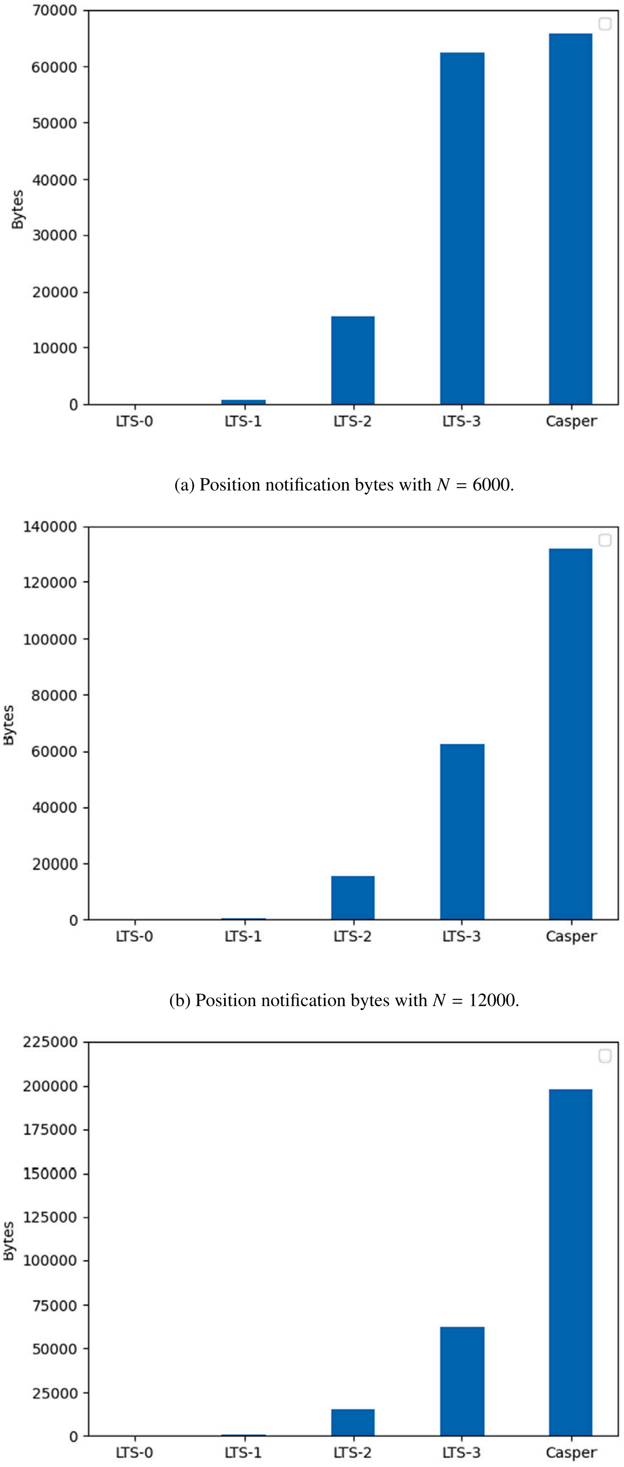
For each value of n, we obtained a different value of x. However, the three values of x are all smaller than 256. This was expected since the size of a cell is 8 m × 8 m and cannot contain a such number of users. Then, less than 1 byte is needed to count the users for each cell. However, to be more conservative, we assume $b = 1$ byte.

The results of our analysis are reported in Figs. 14(a), 14(b), 14(c).

Concerning the impact of the variation of N, our approach is more advantageous than Casper for higher values of N. Indeed, a higher value of N implies that the queries are solved by low-level LTSs.

In all three configurations, the number of bytes received by the LTS of level 3 in our approach is smaller than the number of bytes received by the single LTS employed in Casper. In a high-density scenario, where queries are solved by low-level LTSs, the quantity of bytes received by the LTS of level 3 is approximately one-third of the bytes received by the single LTS employed in Casper. This results in a saving of 66% in overall (non-local) traffic compared to the centralized approach.

In a low-density scenario, our approach keeps more advantageous than the centralized approach, even though the percentage difference decreases to about 5%. Furthermore, the traffic received by low-level LTSs is much smaller than the traffic received by high-level LTSs. Then, by implementing low-level LTSs near the users (at the edge), the quantity of data that flows towards the cloud is reduced. This witnesses the advantage of our decentralized implementation.



(a) Position notification bytes with N = 6000.

(b) Position notification bytes with N = 12000.

(c) Position notification bytes with N = 18000.

Fig. 14. Position notification bytes.

10. Conclusion

In this paper, we tried to solve the issues that a centralized implementation of Location Trusted Service would suffer from. They regard the security and privacy risks associated with the presence of a single point of failure in a system that manages very massive and critical data. Moreover, implementing effective strategies against a global passive adversary would result in huge amount of cover traffic. Consider that the existence of a global observer is not an abstraction because we can expect, on this wide scale, that the location-based provider is one of the big-brother companies, which has a strong tracing power.

Our proposal is to adopt a hierarchical distributed model enabling regional competence of LTSs and a multi-layer service allowing the construction of cloaking areas by combining, in aggregated form, position data coming from different regions. Multiple location-based services are enabled, as one of the elements of a system model for which we also figure out a business model exploiting the modularity of our architecture. Indeed, while a unique centralized LTS can be provided only by very large companies, an LTS with a small territory competence, could be provided also by small or medium enterprises. The proposal overcomes the limitations of the centralized approach, because we split the tracing power among different LTSs and, thus, different independent points. Moreover, by leveraging the edge–cloud paradigm this hierarchical LTS organization allows us to adopt proper mechanisms to resist the global adversary without flooding the entire network of cover traffic.

Our method is parametric with respect to the used cloaking-construction algorithm, also at higher levels of the hierarchy, when the exact positions of the users are not available. The important theoretical result is that if the underlying cloaking-construction-algorithm fulfills the reciprocity property, then our method fulfills this property too, with a low price in terms of effectiveness. This means that the size of the returned cloaking area could be slightly larger than necessary.

To validate our proposal, we conducted an experimental campaign, on a real-life map (our city) on top of a well-known cloaking-construction-algorithm, called Casper.

The validation shows that the price of effectiveness we have to pay is really limited. Specifically, for a wide range of privacy levels ($k > 45$), the experimented percentage difference between the cloaking area size in our approach and Casper was less than 10%. This witnesses that the price in terms of effectiveness is limited.

Moreover, we tested the benefits given by the edge–cloud implementation in terms of network latency, throughput, and cover traffic reduction (to contrast the global passive adversary). Concerning latency and throughput, we obtained an improvement ranging from 20% to 170%, depending on the privacy level k. Moreover, in a high-density scenario, we obtained a saving of 66% of overall (non local) traffic with respect to the centralized approach. In a low-density scenario, our approach keeps more advantageous than the centralized one, even though the percentage difference decreases to about 5%.

We remark that the approach presented in this paper falls in the category of privacy-preserving LBS relying on cloaking areas (and, then, location k-anonymity). As pointed out earlier, this approach aims to protect the identity of the user, thus disarming the power of the attacker to link a victim with a given (sensitive) query, because the author of this query is one among k indistinguishable users. Obviously, if the inclusion of at least k users causes the fact that the cloaking area is sufficiently large (this happens in the case of low-density distribution of people), this approach implicitly applies a certain degree of obfuscation also about the location, which is the objective pursued by techniques such as enlarging. Anyway, it would be interesting to study as a future work how to combine the two classes of methods under our approach. The research question could be the following. Suppose we have, at the lowest level of our hierarchy, the availability of an oracle able to return a cloaking area and an oracle able to return an enlarged region. How to combine these areas to construct, at higher levels of the hierarchy, an approximate area preserving the properties guaranteed by the oracles?

CRediT authorship contribution statement

Francesco Buccafurri: Writing – review & editing, Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Vincenzo De Angelis:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Maria Francesca Idone:** Visualization, Methodology, Investigation, Conceptualization. **Cecilia Labrini:** Writing – original draft, Resources, Methodology, Investigation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was partially supported by the project STRIDE included in the Spoke 5 (Cryptography and Distributed Systems Security) of the Research and Innovation Program PE00000014, “SEcurity and RIghts in the CyberSpace (SERICS)”, under the National Recovery and Resilience Plan, funded by the European Union, NextGenerationEU.

References

[1] C. Bettini, Privacy protection in location-based services: A survey, in: Handbook of Mobile Data Privacy, Springer, 2018, pp. 73–96.

[2] B. Liu, W. Zhou, T. Zhu, L. Gao, Y. Xiang, Location privacy and its applications: A systematic study, IEEE Access 6 (2018) 17606–17624.

[3] J.W. Kim, K. Edemacu, B. Jang, Privacy-preserving mechanisms for location privacy in mobile crowdsensing: A survey, J. Netw. Comput. Appl. (2022) 103315.

[4] I. López-Plata, C. Expósito-Izquierdo, E. Lalla-Ruiz, B. Melián-Batista, J.M. Moreno-Vega, A greedy randomized adaptive search with probabilistic learning for solving the uncapacitated plant cycle location problem, Int. J. Int. Multimedia Artif. Intell. 8 (2) (2023) 123–133, http://dx.doi.org/10.9781/ijimai.2022.04.003, URL: https://www.ijimai.org/journal/sites/default/files/2023-05/ijimai8_2_12.pdf.

[5] S. Özdal Oktay, S. Heitmann, C. Kray, Linking location privacy, digital sovereignty and location-based services: A meta review, J. Locat. Based Serv. (2023) 1–52.

[6] P. Samarati, Protecting respondents identities in microdata release, IEEE Trans. Knowl. Data Eng. 13 (6) (2001) 1010–1027, http://dx.doi.org/10.1109/69.971193.

[7] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, A. Iyengar, Location privacy-preserving mechanisms in location-based services: A comprehensive survey, ACM Comput. Surv. 54 (1) (2021) 1–36.

[8] M. Gruteser, D. Grunwald, Anonymous usage of location-based services through spatial and temporal cloaking, in: Proceedings of the 1st International Conference on Mobile Systems, Applications and Services, 2003, pp. 31–42.

[9] C. Bettini, X.S. Wang, S. Jajodia, Protecting privacy against location-based personal identification, in: W. Jonker, M. Petković (Eds.), Secure Data Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 185–199.

[10] G. Ghinita, K. Zhao, D. Papadias, P. Kalnis, A reciprocal framework for spatial k-anonymity, Inf. Syst. 35 (3) (2010) 299–314.

[11] C.-Y. Chow, M.F. Mokbel, Enabling private continuous queries for revealed user locations, in: International Symposium on Spatial and Temporal Databases, Springer, 2007, pp. 258–275.

[12] C. Anagnostopoulos, S. Hadjiefthymiades, K. Kolomvatsos, Time-optimized user grouping in location based services, Comput. Netw. 81 (2015) 220–244, http://dx.doi.org/10.1016/j.comnet.2015.02.017, URL: https://www.sciencedirect.com/science/article/pii/S1389128615000651.

[13] F. Buccafurri, V. De Angelis, M.F. Idone, C. Labrini, S. Lazzaro, Achieving sender anonymity in Tor against the global passive adversary, Appl. Sci. 12 (1) (2022) http://dx.doi.org/10.3390/app12010137, URL: https://www.mdpi.com/2076-3417/12/1/137.

[14] G. Danezis, C. Diaz, A Survey of Anonymous Communication Channels, Technical Report MSR-TR-2008-35, 2008, p. 46, URL: https://www.microsoft.com/en-us/research/publication/a-survey-of-anonymous-communication-channels/.

[15] F. Buccafurri, V.D. Angelis, S. Lazzaro, Adapting P2P mixnets to provide anonymity for uplink-intensive applications, in: S.D.C. di Vimercati, P. Samarati (Eds.), Proceedings of the 20th International Conference on Security and Cryptography, SECRYPT 2023, Rome, Italy, July 10-12, 2023, SCITEPRESS, 2023, pp. 73–84, http://dx.doi.org/10.5220/0012077100003555.

[16] W.Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, A. Ahmed, Edge computing: A survey, Future Gener. Comput. Syst. 97 (2019) 219–235.

[17] E. ElSalamouny, S. Gambs, Differential privacy models for location-based services, Trans. Data Priv. 9 (1) (2016) 15–48.

[18] S.D.C. di Vimercati, S. Foresti, G. Livraga, P. Samarati, K-anonymity: From theory to applications, Trans. Data Priv. 16 (1) (2023) 25–49.

[19] P. Biswas, A.S. Sairam, Modeling privacy approaches for location based services, Comput. Netw. 140 (2018) 1–14, http://dx.doi.org/10.1016/j.comnet.2018.04.016, URL: https://www.sciencedirect.com/science/article/pii/S1389128618301890.

[20] J. Feng, Y. Wang, J. Wang, F. Ren, Blockchain-based data management and edge-assisted trusted cloaking area construction for location privacy protection in vehicular networks, IEEE Internet Things J. 8 (4) (2020) 2087–2101.

[21] T. Wang, L. Xu, M. Zhang, H. Zhang, G. Zhang, A new privacy protection approach based on K-anonymity for location-based cloud services, J. Circuits Syst. Comput. 31 (05) (2022) 2250083.

[22] J. Domingo-Ferrer, V. Torra, A critique of k-anonymity and some of its enhancements, in: 2008 Third International Conference on Availability, Reliability and Security, IEEE, 2008, pp. 990–993.

[23] Y. Liang, R. Samavi, Optimization-based k-anonymity algorithms, Comput. Secur. 93 (2020) 101753.

[24] L. Kacha, A. Zitouni, M. Djoudi, KAB: A new k-anonymity approach based on black hole algorithm, J. King Saud Univ.-Comput. Inf. Sci. 34 (7) (2022) 4075–4088.

[25] F. Buccafurri, V. De Angelis, S. Lazzaro, Enabling anonymized open-data linkage by authorized parties, J. Inf. Secur. Appl. 74 (2023) 103478, http://dx.doi.org/10.1016/j.jisa.2023.103478, URL: https://www.sciencedirect.com/science/article/pii/S2214212623000625.

[26] L. Xing, X. Jia, J. Gao, H. Wu, A location privacy protection algorithm based on double K-anonymity in the social Internet of Vehicles, IEEE Commun. Lett. (2021).

[27] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, A. Iyengar, Location privacy-preserving mechanisms in location-based services: A comprehensive survey, ACM Comput. Surv. 54 (1) (2021) http://dx.doi.org/10.1145/3423165.

[28] N. Li, T. Li, S. Venkatasubramanian, T-closeness: Privacy beyond k-anonymity and l-diversity, in: 2007 IEEE 23rd International Conference on Data Engineering, IEEE, 2007, pp. 106–115.

[29] M.F. Mokbel, C.-Y. Chow, W.G. Aref, The new casper: Query processing for location services without compromising privacy, in: Proc. of the 32nd International Conf. on Very Large Data Bases, 2006, pp. 763–774.

[30] F. Buccafurri, V. De Angelis, M.F. Idone, C. Labrini, A distributed location trusted service achieving k-anonymity against the global adversary, in: 2021 22nd IEEE International Conference on Mobile Data Management, MDM, IEEE, 2021, pp. 133–138.

[31] A. Ye, Q. Chen, L. Xu, W. Wu, The flexible and privacy-preserving proximity detection in mobile social network, Future Gener. Comput. Syst. 79 (2018) 271–283.

[32] J.W. Kim, K. Edemacu, B. Jang, Privacy-preserving mechanisms for location privacy in mobile crowdsensing: A survey, J. Netw. Comput. Appl. 200 (2022) 103315, http://dx.doi.org/10.1016/j.jnca.2021.103315, URL: https://www.sciencedirect.com/science/article/pii/S1084804521003039.

[33] M.L. Yiu, C.S. Jensen, X. Huang, H. Lu, Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services, in: 2008 IEEE 24th International Conference on Data Engineering, IEEE, 2008, pp. 366–375.

[34] B. Niu, Q. Li, X. Zhu, G. Cao, H. Li, Achieving k-anonymity in privacy-aware location-based services, in: IEEE INFOCOM 2014-IEEE Conference on Computer Communications, IEEE, 2014, pp. 754–762.

[35] B. Wang, J. Liu, L. Dai, K-anonymity-based privacy-preserving and efficient location-based services for internet of vehicles withstand viterbi attack, in: P. You, H. Li, Z. Chen (Eds.), Proceedings of International Conference on Image, Vision and Intelligent Systems 2022, ICIVIS 2022, Springer Nature Singapore, Singapore, 2023, pp. 1016–1028.

[36] S. Zhang, M. Li, W. Liang, V.K.A. Sandor, X. Li, A survey of dummy-based location privacy protection techniques for location-based services, Sensors 22 (16) (2022) http://dx.doi.org/10.3390/s22166141, URL: https://www.mdpi.com/1424-8220/22/16/6141.

[37] M. Wernke, F. Dürr, K. Rothermel, PShare: Ensuring location privacy in non-trusted systems through multi-secret sharing, Pervasive Mob. Comput. 9 (3) (2013) 339–352.

[38] L. Kuang, Y. Wang, X. Zheng, L. Huang, Y. Sheng, Using location semantics to realize personalized road network location privacy protection, EURASIP J. Wireless Commun. Networking 2020 (1) (2020) 1.

[39] D. Parmar, U.P. Rao, Privacy-preserving enhanced dummy-generation technique for location-based services, Concurr. Comput.: Pract. Exper. 35 (2) (2023) e7501.

[40] Y. Fan, L. Liu, X. Zhang, H. Shi, W. Zhai, MAPP: An efficient multi-location task allocation framework with personalized location privacy-protecting in spatial crowdsourcing, Inform. Sci. 619 (2023) 654–678, http://dx.doi.org/10.1016/j.ins.2022.11.075, URL: https://www.sciencedirect.com/science/article/pii/S0020025522013640.

[41] M.L. Damiani, E. Bertino, C. Silvestri, The PROBE framework for the personalized cloaking of private locations, Trans. Data Priv. 3 (2) (2010) 123–148.

[42] C.A. Ardagna, M. Cremonini, S.D.C. di Vimercati, P. Samarati, An obfuscation-based approach for protecting location privacy, IEEE Trans. Dependable Secure Comput. 8 (1) (2009) 13–27.

[43] A. Pingley, N. Zhang, X. Fu, H.-A. Choi, S. Subramaniam, W. Zhao, Protection of query privacy for continuous location based services, in: 2011 Proceedings IEEE INFOCOM, IEEE, 2011, pp. 1710–1718.

[44] Z. Wu, G. Li, S. Shen, X. Lian, E. Chen, G. Xu, Constructing dummy query sequences to protect location privacy and query privacy in location-based services, World Wide Web (2020) 1–25.

[45] Y. Xiao, L. Xiong, Protecting locations with differential privacy under temporal correlations, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 1298–1309.

[46] M. Romanelli, K. Chatzikokolakis, C. Palamidessi, Optimal obfuscation mechanisms via machine learning, in: 2020 IEEE 33rd Computer Security Foundations Symposium, CSF, IEEE Computer Society, 2020, pp. 153–168.

[47] B. NIU, Y. Chen, Z. Wang, B. Wang, H. Li, et al., Eclipse: Preserving differential location privacy against long-term observation attacks, IEEE Trans. Mob. Comput. (2020).

[48] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: An overview, IEEE Signal Process. Mag. 35 (1) (2018) 53–65.

[49] W. Gasarch, A survey on private information retrieval, Bull. EATCS 82 (72–107) (2004) 113.

[50] Y.-B. Zhang, Q.-Y. Zhang, Y. Yan, Y.-L. Jiang, M.-Y. Zhang, A k-anonymous location privacy protection method of polygon based on density distribution, Int. J. Netw. Secur. 23 (1) (2021) 57–66.

[51] S. Siddiqie, A. Mondal, P.K. Reddy, An improved dummy generation approach for infeasible regions, Appl. Intell. (2023) 1–15.

[52] B. Wang, Y. Guo, H. Li, Z. Li, K-anonymity based location privacy protection method for location-based services in Internet of Thing, Concurr. Comput.: Pract. Exper. 35 (20) (2023) e6760, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.6760, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6760.

[53] M. Yang, B. Ye, Y. Chen, T. Li, Y. Yang, X. Qian, X. Yu, A trusted de-swinging k-anonymity scheme for location privacy protection, J. Cloud Comput. 11 (1) (2022) 2.

[54] T. Peng, Q. Liu, G. Wang, Enhanced location privacy preserving scheme in location-based services, IEEE Syst. J. 11 (1) (2014) 219–230.

[55] W. Wang, W. Zhang, Z. Jin, K. Sun, R. Zou, C. Huang, Y. Tian, A novel location privacy protection scheme with generative adversarial network, in: Y. Tian, T. Ma, M.K. Khan (Eds.), Big Data and Security, Springer Singapore, Singapore, 2020, pp. 17–27.

[56] A.K. Gupta, U. Shanker, OMCPR: Optimal mobility aware cache data pre-fetching and replacement policy using spatial K-anonymity for LBS, Wirel. Pers. Commun. (2020) 1–25.

[57] S. Wang, X.S. Wang, In-device spatial cloaking for mobile user privacy assisted by the cloud, in: 2010 Eleventh International Conference on Mobile Data Management, IEEE, 2010, pp. 381–386.

[58] H. Jadallah, Z. Al Aghbari, Spatial cloaking for location-based queries in the cloud, J. Ambient Intell. Humaniz. Comput. 10 (9) (2019) 3339–3347.

[59] J. Lee, S. Kim, Y. Cho, Y. Chung, Y. Park, A hierarchical clustering-based spatial cloaking algorithm for location-based services, J. Internet Technol. 13 (4) (2012) 645–654.

[60] N. Cui, X. Yang, B. Wang, A novel spatial cloaking scheme using hierarchical hilbert curve for location-based services, in: International Conference on Web-Age Information Management, Springer, 2016, pp. 15–27.

[61] G. Zhong, U. Hengartner, A distributed k-anonymity protocol for location privacy, in: 2009 IEEE International Conference on Pervasive Computing and Communications, IEEE, 2009, pp. 1–10.

[62] E. Magkos, P. Kotzanikolaou, S. Sioutas, K. Oikonomou, A distributed privacy-preserving scheme for location-based queries, in: 2010 IEEE International Symposium on "a World of Wireless, Mobile and Multimedia Networks", WoWMoM, IEEE, 2010, pp. 1–6.

[63] G. Ghinita, P. Kalnis, S. Skiadopoulos, Mobihide: A mobilea peer-to-peer system for anonymous location-based queries, in: International Symposium on Spatial and Temporal Databases, Springer, 2007, pp. 221–238.

[64] N. Nisha, I. Natgunanathan, S. Gao, Y. Xiang, A novel privacy protection scheme for location-based services using collaborative caching, Comput. Netw. 213 (2022) 109107, http://dx.doi.org/10.1016/j.comnet.2022.109107, URL: https://www.sciencedirect.com/science/article/pii/S1389128622002377.

[65] S. Zhang, K.-K.R. Choo, Q. Liu, G. Wang, Enhancing privacy through uniform grid and caching in location-based services, Future Gener. Comput. Syst. 86 (2018) 881–892.

[66] A.P. Mdee, M.M. Saad, M. Khan, M.T.R. Khan, D. Kim, Impacts of location-privacy preserving schemes on vehicular applications, Veh. Commun. 36 (2022) 100499, http://dx.doi.org/10.1016/j.vehcom.2022.100499, URL: https://www.sciencedirect.com/science/article/pii/S2214209622000468.

[67] Z. Lu, G. Qu, Z. Liu, A survey on recent advances in vehicular network security, trust, and privacy, IEEE Trans. Intell. Transp. Syst. 20 (2) (2019) 760–776, http://dx.doi.org/10.1109/TITS.2018.2818888.

[68] M. Tao, W. Wei, S. Huang, Location-based trustworthy services recommendation in cooperative-communication-enabled internet of vehicles, J. Netw. Comput. Appl. 126 (2019) 1–11, http://dx.doi.org/10.1016/j.jnca.2018.10.023, URL: https://www.sciencedirect.com/science/article/pii/S1084804518303515.

[69] X. Li, Y. Ren, L.T. Yang, N. Zhang, B. Luo, J. Weng, X. Liu, Perturbation-hidden: Enhancement of vehicular privacy for location-based services in internet of vehicles, IEEE Trans. Netw. Sci. Eng. 8 (3) (2021) 2073–2086, http://dx.doi.org/10.1109/TNSE.2020.3011607.

[70] B. Brecht, D. Therriault, A. Weimerskirch, W. Whyte, V. Kumar, T. Hehn, R. Goudy, A security credential management system for V2X communications, IEEE Trans. Intell. Transp. Syst. 19 (12) (2018) 3850–3871.

[71] C.-W. Chen, S.-Y. Chang, Y.-C. Hu, Y.-W. Chen, Protecting vehicular networks privacy in the presence of a single adversarial authority, in: 2017 IEEE Conference on Communications and Network Security, CNS, 2017, pp. 1–9, http://dx.doi.org/10.1109/CNS.2017.8228648.

[72] F. Schaub, F. Kargl, Z. Ma, M. Weber, V-tokens for conditional pseudonymity in VANETs, in: 2010 IEEE Wireless Communication and Networking Conference, IEEE, 2010, pp. 1–6.

[73] H. Shen, J. Zhou, Z. Cao, X. Dong, K.-K.R. Choo, Blockchain-based lightweight certificate authority for efficient privacy-preserving location-based service in vehicular social networks, IEEE Internet Things J. 7 (7) (2020) 6610–6622, http://dx.doi.org/10.1109/JIOT.2020.2974874.

[74] K. Bala, R. Upadhyay, S.R. Anwar, G. Shrimal, A blockchain-enabled, trust and location dependent-privacy preserving system in VANET, Measurement: Sensors 30 (2023) 100892.

[75] V.K. Yadav, N. Andola, S. Verma, S. Venkatesan, Anonymous and linkable location-based services, IEEE Trans. Veh. Technol. 71 (9) (2022) 9397–9409, http://dx.doi.org/10.1109/TVT.2022.3180412.

[76] B. Ma, X. Lin, X. Wang, B. Liu, Y. He, W. Ni, R.P. Liu, New cloaking region obfuscation for road network-indistinguishability and location privacy, in: Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses, 2022, pp. 160–170.

[77] M. Li, Y. Chen, N. Kumar, C. Lal, M. Conti, M. Alazab, Quantifying location privacy for navigation services in sustainable vehicular networks, IEEE Trans. Green Commun. Netw. 6 (3) (2022) 1267–1275, http://dx.doi.org/10.1109/TGCN.2022.3144641.

[78] T. Brinkhoff, A framework for generating network-based moving objects, GeoInformatica 6 (2) (2002) 153–180.

[79] L. Kissner, D. Song, Privacy-preserving set operations, in: Annual International Cryptology Conference, Springer, 2005, pp. 241–257.

[80] S.K. Debnath, P. Stănică, N. Kundu, T. Choudhury, Secure and efficient multi-party private set intersection cardinality, Adv. Math. Commun. 15 (2) (2021) 365.

[81] C. Dong, G. Loukides, Approximating private set union/intersection cardinality with logarithmic complexity, IEEE Trans. Inf. Forensics Secur. 12 (11) (2017) 2792–2806.

[82] M. Masdari, S. Gharehpasha, M. Ghobaei-Arani, V. Ghasemi, Bio-inspired virtual machine placement schemes in cloud computing environment: Taxonomy, review, and future research directions, Cluster Comput. 23 (4) (2020) 2533–2563.

[83] M. Reiss-Mirzaei, M. Ghobaei-Arani, L. Esmaeili, A review on the edge caching mechanisms in the mobile edge computing: A social-aware perspective, Internet Things 22 (2023) 100690, http://dx.doi.org/10.1016/j.iot.2023.100690, URL: https://www.sciencedirect.com/science/article/pii/S2542660523000136.

[84] L. Tong, Y. Li, W. Gao, A hierarchical edge cloud architecture for mobile computing, in: IEEE INFOCOM 2016-the 35th Annual IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.

[85] J. Henriques, F. Caldeira, A model for planning TELCO work-field activities enabled by genetic and ant colony algorithms, Int. J. Interact. Multimedia Artif. Intell. 7 (6) (2022) 24–30, http://dx.doi.org/10.9781/ijimai.2022.08.011, URL: https://www.ijimai.org/journal/sites/default/files/2022-09/ijimai7_6_3.pdf.

[86] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1657–1681.

[87] A. Joyce, R.L. Paquin, The triple layered business model canvas: A tool to design more sustainable business models, J. Clean. Prod. 135 (2016) 1474–1486.

[88] F. Buccafurri, V. De Angelis, M.F. Idone, C. Labrini, A protocol for anonymous short communications in social networks and its application to proximity-based services, Online Soc. Netw. Media 31 (2022) 100221, http://dx.doi.org/10.1016/j.osnem.2022.100221, URL: https://www.sciencedirect.com/science/article/pii/S2468696422000258.

[89] J. Wang, Casper-cloaking implementation in JAVA, 2018, GitHub repository GitHub, https://github.com/iwangjian/Casper-Cloaking.

[90] R. Yu, Z. Bai, L. Yang, P. Wang, A. Oguti, Y. Liu, A location cloaking algorithm based on combinatorial optimization for location-based services in 5G networks, IEEE Access 4 (2016) 6515–6527, http://dx.doi.org/10.1109/ACCESS.2016.2607766.

[91] M. Orabi, Z. Al Aghbari, I. Kamel, FogLBS: Utilizing fog computing for providing mobile location-based services to mobile customers, Pervasive Mob. Comput. 94 (2023) 101832, http://dx.doi.org/10.1016/j.pmcj.2023.101832, URL: https://www.sciencedirect.com/science/article/pii/S1574119223000901.

[92] M. Haklay, P. Weber, Openstreetmap: User-generated street maps, IEEE Pervas. Comput. 7 (4) (2008) 12–18.

[93] H. Wang, Z. Zhang, T. Wang, S. He, M. Backes, J. Chen, Y. Zhang, PrivTrace: Differentially private trajectory synthesis by adaptive Markov model, in: USENIX Security Symposium 2023, 2023.

[94] P. Mishra, U. Singh, C.M. Pandey, P. Mishra, G. Pandey, Application of student's t-test, analysis of variance, and covariance, Ann. Cardiac Anaesthesia 22 (4) (2019) 407.

[95] L. Davisson, J. Jakovleski, N. Ngo, C. Pham, J. Sommers, Reassessing the constancy of end-to-end internet latency, Traffic 41 (44) (2021) 46.

[96] F. Buccafurri, V. de Angelis, S. Lazzaro, MQTT-A: A broker-bridging P2P architecture to achieve anonymity in MQTT, IEEE Internet Things J. 10 (17) (2023) 15443–15463, http://dx.doi.org/10.1109/JIOT.2023.3264019.

[97] Digital Ocean, Digital ocean cloud platform, 2023, https://docs.digitalocean.com/. (Accessed: 25 February 2024).

[98] K. Nath, System Design: Design a Geo-Spatial index for real-time location search, 2023, https://kousiknath.medium.com/system-design-design-a-geo-spatial-index-for-real-time.location-search-10968fe62b9c. (Accessed: 25 February 2024).

[99] R. Stoleru, T. He, J.A. Stankovic, Walking GPS: A practical solution for localization in manually deployed wireless sensor networks, in: 29th Annual IEEE International Conference on Local Computer Networks, IEEE, 2004, pp. 480–489.

Francesco Buccafurri is a full professor of computer science at the University Mediterranea of Reggio Calabria, Italy. In 1995 he took the Ph.D. degree in computer science at the University of Calabria. In 1996 he was visiting researcher at the database and knowledge representation group of Vienna University of Technology. His research interests include cybersecurity and privacy, social networks, deductive-databases, knowledge-representation and non-monotonic reasoning, model checking, data compression, data streams, agents, P2P systems. He has published more than 160 papers in top-level international journals and conference proceedings. He serves as a referee for international journals and he is a member of a number of conference PCs. Francesco Buccafurri is Associate Editor of Information Sciences (Elsevier) and IEEE Transactions on Industrial Informatics, he is included in the editorial board of a number of other international journals, and played the role of PC chair and PC member in many international conferences. He is member of the IEEE computer society.

Vincenzo De Angelis is assistant professor at the University of Calabria, Italy. In 2023 he took the Ph.D. degree at the University of Reggio Calabria. His research interests include IoT security, blockchain, cloud, and applied cryptography. He is author of more than 20 papers published in international journals and conference proceedings. He is PC member of a number of conferences and Guest Editor of a special issue in an international Journal.

Maria Francesca Idone received the Master's degree in telecommunication engineering in 2020. Her research interests include information security, privacy, and social networks. She is author of a number of papers published in international conference proceedings.

Cecilia Labrini received the Ph.D. in information engineering at the University Mediterranea of Reggio Calabria, Italy in 2023. She received the Master's degree in telecommunication engineering in 2019. Her research interests include information security, blockchain, cloud, and trust propagation. She is author of a number of papers published in international conference proceedings. She is PC member of some conferences. She is participating in several research projects.