Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at *Sensors* **2018**, *19*, 104; doi:10.3390/s19010104

Article

Fog architectures and sensor location certification in distributed event-based systems

Fátima Castro-Jul [1]*, Rebeca P. Díaz-Redondo [1], Ana Fernández-Vilas [1], Sophie Chabridon [2] and Denis Conan [2]

[1] I&C Lab, AtlantTIC Research Center, Universidade de Vigo, Spain; {fatima,rebeca,avilas}@det.uvigo.es
[2] SAMOVAR, Télécom SudParis, CNRS, Université Paris-Saclay, Évry, France; {firstname.lastname}@telecom-sudparis.eu
* Correspondence: fatima@det.uvigo.es; Tel.: +34 986818704

Abstract: Since smart cities aim at becoming self-monitoring and self-response systems, their deployment relies on close resource monitoring through large-scale urban sensing. The subsequent gathering of massive amounts of data makes essential the development of event filtering mechanisms that enable the selection of what is relevant and trustworthy. Due to the rise of mobile event producers, location information has become a valuable filtering criterion as it not only offers extra information on the event described but also enhances trust on the producer. Implementing mechanisms that validate the quality of location information becomes then imperative. The lack of such strategies in cloud architectures compels the adoption of new communication schemes for IoT-based urban services. To serve the demand for location verification in urban event-based systems (DEBS), we have designed three different fog architectures that combine proximity and cloud communication. Moreover, we have successfully assessed their performance using network simulations with realistic urban traces.

Keywords: participatory sensing, smart cities, Internet of Things, distributed event-based systems

1. Introduction

Smart cities are intelligent and interconnected environments where resources are constantly monitored in order to, among other objectives, assess their performance and ensure an appropriate reaction in case of an incident. From transportation to safety or energy management, the list of services that can benefit from obtaining real-time information to enhance citizens' experience is almost endless [1]. Monitoring a whole city implies extensive sensing and gathering large amounts of data. The emergence of the Internet of Things (IoT) eases this task by enabling the interconnection of sensors embedded in any device, vehicle or object.

Since they provide time and space decoupling, distributed event-based systems (DEBS) [2] enable flexible communication in dynamic and heterogeneous urban scenarios. As a result, they have become a useful interaction mechanism for IoT-based smart city services [3]. Moreover, they support event filtering, data aggregation, and scalability, which are essential in the development of large-scale IoT-based services [4] . Event filtering is of paramount importance in large-scale urban sensing, where an enormous quantity of data is generated. By choosing their event subscriptions, consumers can select the most relevant and trustworthy data available to improve their environment awareness in a reliable way. Therefore, event filtering is not limited to the event notification content but also extends to its quality, evaluated using additional knowledge provided by event producers and event processing agents [5].

When considering filtering criteria, location is one of the first that come to mind. The inclusion of geographical information on event notifications allows subscriptions based on areas of interest. Thus,

© 2018 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

it helps consumers to select notifications that are relevant for them. Additionally, it reinforces trust in producers. Even though location information may not be significant in every notification, the fact that producers provide extra information about their whereabouts increases their reliability. As a result, implementing location evaluation mechanisms improves trust both on the event notification source and on the notification itself.

Location assessment mechanisms differ according to the producers' mobility and the DEBS architecture. Static event producers are known to be in a certain location and, therefore, it is easy to detect fake location claims coming from them. However, when it comes to mobile producers, location assessment becomes more difficult. The importance of mobile producers has considerably grown over the last ten years, due to the emergence of the participatory sensing [6] and mobile crowdsensing paradigms [7]. Producers that move through the city and gather information about different areas are a promising alternative to building a costly fixed sensing and communication infrastructure to cover a whole city. As a result, mobile clients and the subsequent location issues need to be considered when designing urban DEBS. In classical cloud architectures, an overlay of brokers in the cloud lets producers and consumers communicate, placing a blind confidence in the locations they claim. A possible solution is to switch from this architecture to a proximity-based totally ad hoc network that ensures producers communication directly with consumers in their very area [8]. This is a good solution for small and densely populated urban areas but it becomes expensive when producers and their subscribed consumers are far away from each other. Another alternative is to migrate part of the data processing from the cloud to the edges of the network. Thus, location is verified on the edges, in the devices' proximity, while a cloud architecture for wide-spread area communication is maintained. Extending cloud architectures closer to the users is not a novel idea, it is the basis of the Fog Computing paradigm [9,10].

The capabilities of fog architectures for sensor location certification in urban DEBS remain largely unexplored. In order to evaluate their potential, we have designed and assessed through simulation three different architectures based on fog computing that combine cloud and proximity communication. The architectures are aimed at improving the quality of urban notifications and enable wide area communication while benefiting from proximity-based location verification on network edges. To do so, we enrich traditional cloud architectures with short-range communication schemes that enable peer-based location verification. This proposal evolves from our works on quality of context information in DEBS [11] and on distributed proximity-based collaboration with peer devices [8]. The architectures, albeit briefly presented in our previous work [12], are extensively studied in this paper. Moreover, their performance is assessed using network simulations with realistic urban traces.

This paper is organized as follows. First, Section 2 presents an overview of how location verification is dealt with in urban sensing architectures. Then, Section 3 analyses the scenario our approach is targeted at, and Section 4 outlines the basics of our DEBS system. Our proposed architectures are described in Section 5 while their simulation and evaluation are discussed in Sections 6 and 7. Finally, a conclusion and some guidelines for future work can be found in Section 9.

2. Background

The high potential of mobile sensor-enabled devices as a powerful tool to monitor their surroundings has led to the emergence of participatory [6] and crowdsensing systems [7,13], which leverage user-generated information to form collaborative knowledge about urban areas. These systems enhance the quality and credibility of urban sensing and therefore, have become a major player in IoT distributed event-based platforms [14]. However, the prevalence of mobile event producers poses new challenges in urban sensing schemes.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

On the one hand, the possibility of exploiting an ever-growing number of mobile devices results in a massive increase in the quantity of data collected. As a result, developing filtering mechanisms becomes imperative. The focus of those may be reducing the transmission of unnecessary information on the publishing side [3,8,14] or supporting selective subscriptions on the consumming side, based on criteria such as trust on the producer and information quality [11].
On the other hand, sensing platforms need to handle the inherent device mobility. This includes dealing with an unequal area coverage due to human mobility patterns [3], translating information on producers among different locations [15], and analyzing location claims that are continuously changing. Since sensed information is often geotagged for contextualization, verifying location claims is of paramount importance to ensure adequate subscription management and to reinforce trust in the sensing system. Thus, the location verification problem relates to both of the mentioned challenges in participatory sensing: it arises from the high mobility of producers and, considering that location is a major filtering criterion, it significantly affects effective data dissemination. The task of designing participatory DEBS with location verification support varies depending on the different system architectures, whose main characteristics are described next.
DEBS are typically based on three components [2]: producers (P), consumers (C) and brokers (B). Producers advertise their contribution to the system, and then publish events that match their advertisement. Consumers subscribe to events of their interest and receive them. Brokers handle consumer subscriptions in order to route event notifications appropriately. Every client (consumer or producer) is connected to at most one broker at the same time. We distinguish between three different system architectures: cloud (Section 2.1), cloudless (Section 2.2) and fog (Section 2.3).

2.1. Cloud architectures

Traditional cloud architectures [3,14] consist of an overlay of brokers in the cloud that communicates producers and consumers located anywhere. Figure 1.a depicts a cloud-based participatory sensing with producers (P_i) and consumers (C_j) in different city sections. There is also an overlay of interconnected brokers (B_k) that allows communication over the whole area and whose internal organization is transparent to the clients. The clients (producers and consumers) connect first to a central server, which assigns them to an access broker. The access broker can be anywhere and may serve other clients from different locations. It does not know the clients' position and has no means to verify whether they are claiming false locations. As a result, the access broker places blind trust in the locations submitted by producers.

2.2. Cloudless architectures

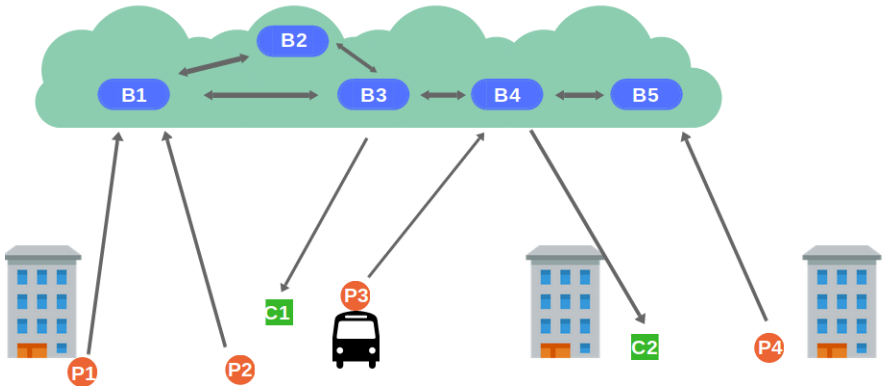
Moving the whole sensing system to the clients' proximity ensures producers communicate only with those in their actual whereabouts. By dispensing with the cloud and relying solely on short-range communication technologies, false location claims become easily detectable. Cloudless architectures may consist in a broker infrastructure placed all over the urban area that needs to be covered (Figure 1.b). Access brokers (B_k) are always near their clients and therefore, can verify whether their location claims are accurate. Brokerless or ad hoc architectures are another alternative (Figure 1.c). Clients (P_i and C_j) communicate directly with each other and form opportunistic networks to deliver event notifications to those interested [8]. In this scenario, every client takes part in the verification of notifications created in their proximity. Both the architectures work well in small and, in the ad hoc case, densely-populated zones. However, they are costly to deploy in wide areas, and do not enable communication with clients outside of it.

2.3. Fog architectures

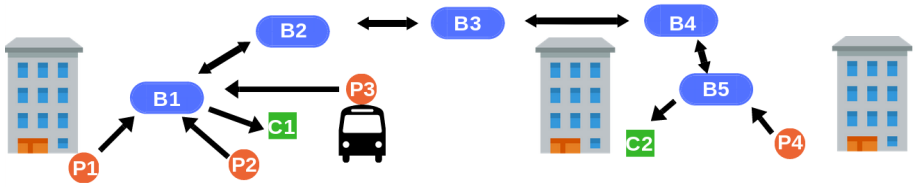
The Fog Computing paradigm advocates the establishment of an intermediate layer between end devices and traditional cloud servers that provides computation, storage and networking

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104



(a) Cloud architecture



(b) Cloudless infrastructure architecture

(c) Ad hoc architecture

Figure 1. Participatory sensing architectures

services [9,10]. It has emerged as a useful tool for IoT architectures since it provides mobility and heterogeneity support [16], low latency and scalability [17]. Moving part of the data processing closer to the ground increases the system awareness about clients' locations. Moreover, since part of the architecture is still in the cloud, wide-spread area coverage is guaranteed. There exist different alternatives in the design of fog computing architectures, where the intermediate layer can be placed either remotely or physically.

2.4. Our contribution

To the best of our knowledge, there exist no distributed solutions for location verification support in event-based sensing systems. The lack of such mechanisms prevents the implementation of reliable location-based subscriptions and reduces the confidence consumers place in event producers and thus, in the system itself. As a consequence, the development of urban services based on information gathering through participatory event-based schemes is compromised. Hence, it is imperative to analyze the potential of fog architectures for location verification in DEBS.

Our contribution focuses on employing proximity-based communication to verify producers' location. Proximity-based communication connects devices that are placed in the same area, at a distance from

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

each other that allows them to be reached using a wireless short-range communication technology, such as Bluetooth or ad hoc WiFi. This kind of connection diminishes the possibility of producers to fake their location, unless just for a few meters. If a producer that claims to be in a particular area cannot be reached by other devices known to be there, we can conclude that it is not providing its actual location. Following this principle, we have designed three different DEBS architectures that support location verification: the fixed brokers architecture, the assigned brokers architecture and the collaborative architecture. Before detailing our location verification proposal and introducing the architectures, we present in the next section the scenario we use for illustrative purposes.

3. Scenario

We have targeted our scenario at a participatory sensing DEBS with high producer mobility and periodic sensing data publications. We have chosen an environmental variable that can be detected merely using a smartphone and presents multiple applications: noise monitoring. Direct measurements are employed for noise pollution assessment, which is a topic of concern due to its consequences on citizens' health and wellbeing. Moreover, noise measurements can lead to the inference of different events such as traffic jams, road works or crowds. A pub/sub system where users can receive geolocated noise data helps them to avoid highly-polluted or congested areas of the city by choosing alternative routes. Besides, this kind of system serves as a detection mechanism for events such as concerts or demonstrations. In order to provide consumers with precise event positioning, it is essential to provide reliable location data.

To avoid the high cost of city-wide infrastructure deployment, noise measurements can be performed using mobile sensors carried by vehicles or users. The latter is especially interesting since smartphones' microphones have been proved to successfully act as noise sensors [18]. Therefore, no dedicated infrastructure is necessary at all. Having mobile devices engaged in dynamic and interactive networks to share local information makes this a participatory sensing scenario [6].

Noise measurements are automatically issued every few minutes. Thus, possible events can be quickly detected, and subscribers receive fresh data frequently. Using a fog architecture allows users to receive noise measurement event notifications anywhere. Therefore, they can, for instance, be aware of incidents in their place of residence even when they are away or discover if an area in a different city they want to move or travel to is noisy.

The same system can be adapted to other urban variables such as air pollution.

4. DEBS with location verification support

Our goal is to improve event notification quality by increasing trust in the event's source location. In order to focus our attention on producers' mobility, we assume the broker and consumer structure to be fixed. Producers advertise locally and consumers subscribe globally. Thus, the broker network is always aware of the subscription needs and communicates consumers with the appropriate producers. Simple routing is employed: each subscription leads to the installation of a direct spanning tree that routes notifications towards the consumer.

Location information is inserted as an attribute in every event notification. If it can be verified, notifications also include a location certification. The location attribute can be verified either on the producer or on the broker, but it is always the broker, assumed to be trustworthy, the one that decides whether to provide the certification. Consumers may choose to subscribe only to events with certified location. The verification process consists on assessing the location attribute using either extra information provided by the producer to support the location claim or the collaboration of other peers in the area. The certification process consists on adding a location certificate in the event notification to state the veracity of the location claim. An event notification is certified if the location claim has been verified. The verification and certification processes ensure that at least one agent has reviewed the location attribute provided by the producer and assessed it. As a result, trust in the notification source is increased.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

In this section, we describe the distinct characteristics of a DEBS with location verification support: the role of producers and brokers in the verification, the definition of the location attribute and the management of mobility. Moreover, we indicate how some of the features differ according to the three architectures we propose in this paper, which are described in detail in Section 5: the fixed brokers architecture, the assigned brokers architecture and the collaborative architecture.

4.1. Producers' role

For the sake of simplicity and without impairing generality, notifications contain attributes as sets of triples: ($n_{1,i}$, $n_{2,i}$, v_i), where $n_{1,i}$ and $n_{2,i}$ are strings that uniquely identify the attribute and v_i is the value of the attribute. For instance, the location attribute is inserted in the notification as the triple (location, value, X)[1], where X is a variable that represents the value of the location. Producers include this latter triple in every notification. Moreover, the publishing process may also involve the verification of the location attribute and the incorporation of extra information to assist the broker in the verification process. In Algorithm 1, function *PrepareExtraLocationInformation* deals with the tasks of verifying location and arranging extra location data, in case they are considered. It returns the arranged information, which function *PrepareNotification* inserts in the notification, together with the location triple. Both these functions vary their operation according to the architectures. Their different definitions are presented in Section 5.
Since a producer can disconnect and reconnect, its architecture is supplemented with a queue of notifications where they are stored when no access broker is reachable. Queued notifications are transmitted to the access broker the client connects next.

Algorithm 1 : at producer P, publish

```
1: procedure PUBLISH(Notification n)
2:     lc ← PrepareExtraLocationInformation()
3:     n ← PrepareNotification(n,lc)        ▷ Adds location value and extra information, if any
4:     if connected to an access broker then
5:         send n to the access broker
6:     else
7:         store n in the local queue
8:     end if
9: end procedure
```

4.2. The location attribute

The location attribute may consist of an explicit representation using coordinates or of an identifier of a specific area, neighborhood or city. The former supports more precise location data but implies more expensive event processing. Considering that the areas are delimited by a short-range communication technology, we consider that area identifiers are precise enough in this work. We assume the existence of a location ontology, known by all the agents in the system. Moreover, producers incorporate a geolocation mechanism that allows them to identify their own position and describe it with a valid location value.
The location ontology can include a hierarchical structure that allows the definition of different levels of location granularity and notification visibility.

4.3. Brokers' role

Brokers act not only as notification routers but also involve event processing agents. A common one for the three proposed architectures is a translate event processing agent [19], in charge of enriching the verified notifications with a location certificate in the form of the attribute triple:

[1] Since an attribute can be associated not only with its value but also with metadata, it may be described by multiple triples, e.g., (location, value, X), (location, certified, Y).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

(location, certified, Y), where the first two elements describe the attribute and Y can take two possible values: {*true*, *false*}. To establish the veracity of the location claim, the broker may rely on the extra information provided by the producer in the notification. The nature and the treatment of this information vary in the different architectures. In the first two architectures (fixed brokers architecture, assigned brokers architecture), there exists a filtering agent that identifies the notifications whose location is the same as the broker's one. Additionally, brokers in the assigned brokers architecture and the collaborative architecture integrate an agent whose role is to check the information provided by the producers to support their location claim and to remove it from the notification. The function *VerifyLocation* is in charge of processing the extra information, removing it from the notification and returning a value that indicates whether the location claim has been verified or not. The definition of the function differs according to the different architectures, as detailed in Section 5.
Algorithm 2 depicts the notification handling process.

Algorithm 2 : at broker B, handle local notifications

```
1:  procedure HANDLELOCALNOTIFICATION(Producer P, Notification n)
2:      if B is the access broker of P then
3:          locOk ← VerifyLocation(n)
4:          if locOk then
5:              insert triple (location, certified, true)
6:          else
7:              insert triple (location, certified, false)
8:          end if
9:      end if
10:     Forward n to interested neighbor brokers and local consumers
11: end procedure
```

4.4. Mobility

When brokers are assigned to geographical areas, a change in physical location also implies a change in the broker connection. Regardless of their location, producers publish periodic notifications, which are processed by the access broker they are connected to at that time. Thus, mobility handling is transparent for the producers. This concept of mobility corresponds to the physical mobility defined by Fiege et al. [20]. In the first two proposed architectures, mobility is handled by the local broker, installed on every client. It is in charge of establishing connections with border brokers and delivering them the notifications. It becomes aware of a location change when a broker connection is no longer available.
In the fixed brokers architecture, this happens when the producer moves out of the reach of the broker communication sphere. Then, the producer listens for beacons from another border broker to connect to. In the assigned brokers architecture, the connection is canceled when the producer is no longer able to send valid periodic notifications. At that point, the producer becomes aware of the location change and sends a new connection request with the producer's location coordinates.
Event notifications published by disconnected producers are queued locally. When a new connection is established, they are issued to the new border broker. If the location attribute of the queued notifications is different from the new broker's location, the notifications will be distributed but it will not be certified because a broker cannot certify a location different from its own.
The situation is different in the collaborative architecture. There, changes in broker connection do not imply location changes and vice versa. Since brokers are not assigned to a certain area, there are no limitations on the notifications they can consider for certification. As long as the location meets the criteria to be considered collaboratively decided (defined in Section 5.3), it can always be certified regardless of changes in the broker connection or the producer location.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

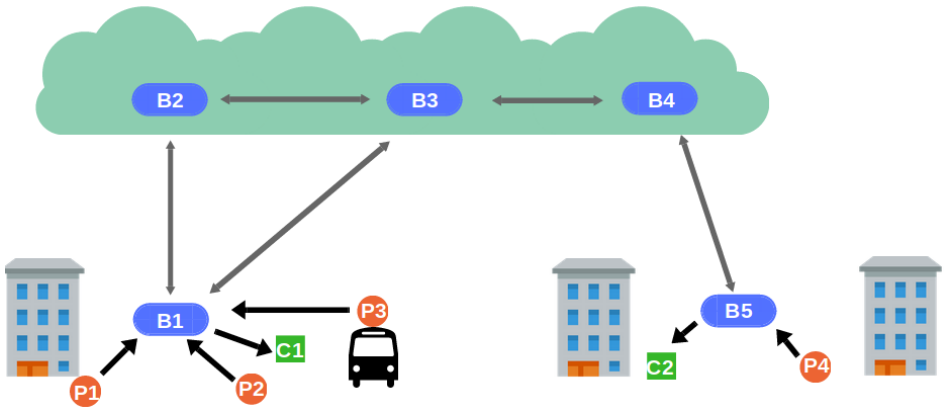


Figure 2. Fixed brokers architecture

5. Architectures

In this section, we present three different fog architectures for a DEBS system with location verification support and we outline their main features.

5.1. Fixed brokers architecture

The first architecture is shown in Figure 2. There exist two levels of brokers: border brokers (B1, B5) and inner brokers (B2, B3, B4). While inner brokers are in the cloud, border brokers are placed around the city covering different areas. They are fixed and, therefore, they are assigned a fixed location value. Access brokers are connected to inner brokers to enable event distribution. Clients cannot directly interact with inner brokers, whose overlay structure is transparent to them. Clients in the same area are connected to the same access broker using a short-range transmission technology that ensures the veracity of the clients' location. Thereby, in this architecture location verification is straightforward. There is no verification on the producer (Algorithm 3), and the access broker will certify the location of every event notification whose location attribute coincides with its own location (Algorithm 4).

Algorithm 3 : at producer P, prepare extra location information (Fixed brokers architecture)

```
1: function PrepareExtraLocationInformation
2:     return ∅
3: end function
```

In this architecture, connections are simple. Brokers advertise themselves through beaconing using a short-range communication technology. When a producer that is not connected to any broker is reached by a beacon, it establishes a connection with the sending broker. Since the communication is performed using a short-range communication technology, there is no need for further verification to prove the producer is in the broker's area. The producer and the broker exchange periodic messages to maintain the connection. When they stop receiving these messages, they consider the connection as broken. The broker deletes the producer's advertisements and the producer starts paying attention to beacons to establish a connection with another broker.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

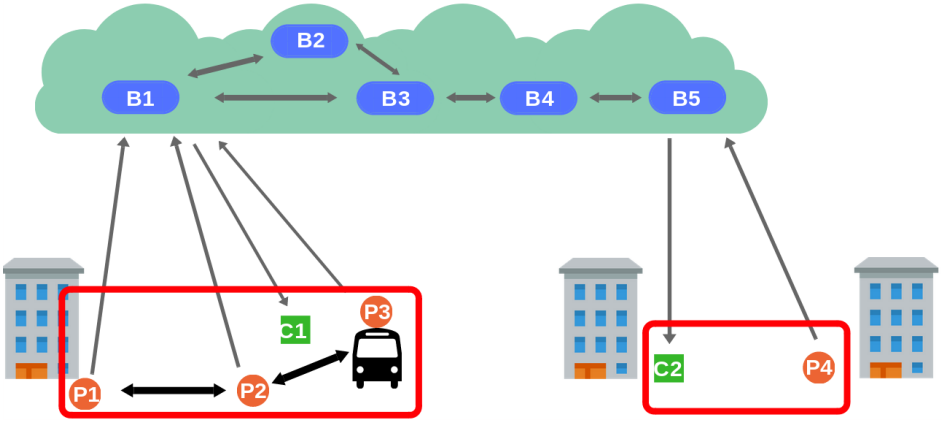


Figure 3. Assigned brokers architecture

Algorithm 4 : at broker B, verify location (Fixed brokers architecture)

```
1: function VERIFYLOCATION(Notification n)
2:     location ← {Y|{location, value, Y} ∈ n}
3:     if location == B's location then
4:         return true
5:     else
6:         return false
7:     end if
8: end function
```

5.2. Assigned brokers architecture

Having a hybrid architecture that combines fixed and mobile agents diminishes the infrastructure requirements of deploying a purely fixed sensor architecture. However, the cost of setting up fixed brokers may still be high. As a result, we have designed a second architecture, which also exploits location verification through proximity-based communication but gives up the requirements for fixed infrastructure. Figure 3 displays the same DEBS network shown in Figure 2 but with all the brokers in the cloud. By considering location-based network partitions, we can organize client connections to brokers such as to maximize the probability of having all the clients in an area connected to the same broker. Then, brokers can still be in charge of producers' location verification even though they cannot communicate with them using a short-range transmission technology. Each broker is assigned an area and maintains a list of registered producers, updated with every publication received. Producers obtain the IP address of their access brokers by providing their location to a discovery service. To maintain the connection, producers are required to send valid notifications to the broker periodically. When a producer sends a notification, it includes the list of other producers in the area with whom it has established a short-range communication link and who are connected to the same broker (Algorithm 5). Producers claiming a false location are not able to provide a valid neighbor list. Then, the broker compares the neighbor list provided in the received notification with the list of devices in the area it maintains. The publication is certified if it satisfies the requirements detailed below. If the publication is certified, the sender is included in the list of registered producers in that area.

Algorithm 5 : at producer P, prepare extra location information (Assigned brokers architecture)

```
1: function PREPAREEXTRALOCATIONINFORMATION
2:     return neighborList
3: end function
```

Relying solely on mobile producers involves extra challenges. Producers on area edges are likely to be reached by beacons from producers in a different area. Therefore, information on neighbors from

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

nearby regions need to be considered when assessing the location of every publication. As a result, every broker should exchange its list of registered producers with the brokers assigned to contiguous areas.

5.2.1. Verification strategies

Two different verification strategies are proposed: the complete-list strategy (CLS) and the non-empty list strategy (NLS). In the first one, the neighbor list provided by the producer needs to satisfy two requirements. First, it has to include every producer registered with the broker assigned to that area. This condition requires areas small enough for producers to find every one of their peers registered to the same broker with high probability.
Second, the list cannot include extra neighbors that are neither registered with the broker nor with the brokers assigned to adjacent areas. Otherwise, producers may include a neighbor list with as many producer identifiers as possible in the hope that some of them are actually registered with the broker. These requirements can be written as:

1. $\forall b \in P_B, b \in NP$
2. $\forall b \in NP, b \in P_B \cup P_{NB}$

where NP is the list of neighbors provided by the producer, P_B is the list of producers registered with the broker assigned to that area and P_{NB} is the list of producers registered with brokers assigned to contiguous areas.
CLS is not effective in situations where the producer has not found any neighbors and there are no producers registered with the broker either, i.e., both NP and P_B are empty. In this case, the broker certifies the notification regardless of the veracity of the location claim.
To resolve this issue, we propose a second verification strategy (NLS), where the producer is required to provide a neighbor that is registered with the broker — i.e., $b \in NP \cup P_B$. However, this strategy is not optimal either. It could lead to the certification of notifications with false location claims that include a long list of producers, whereof one happens to be registered with the broker. This is problematic in scenarios where a producer can maintain a list of every producer it has ever known and use it as its current neighbor list.

Algorithm 6 : at broker B, verify location (Assigned brokers architecture, CLS)

```
1:  function VerifyLocation(Notification n)
2:      NP ← neighborList ∈ n
3:      remove neighborList from n
4:      location ← {Y|{location, value, Y} ∈ n}
5:      if location ==B's location ∧(∀b ∈ P_B, b ∈ NP) ∧ (∀b ∈ NP, b ∈ P_B ∪ P_NB) then
6:          return true
7:      else
8:          return false
9:      end if
10: end function
```

As a result, CLS is better suited to scenarios with a high device density where there are producers in every area and, therefore, assigned to every broker. NLS is targeted at scenarios where the density of devices is low and where no producer is assumed to have global knowledge about peers in the whole system. We include both the strategies in our simulation analysis.
Algorithms 6 and 7 describe the location verification procedure in the broker for CLS and NLS respectively.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

Algorithm 7 : at broker B, verify location (Assigned brokers architecture, NLS)

```
1:  function VERIFYLOCATION(Notification n)
2:      NP ← neighborList ∈ n
3:      remove neighborList from n
4:      location ← {Y|{location, value, Y} ∈ n}
5:      if location ==B's location ∧(∃b ∈ NP, b ∈ P_B) then
6:          return true
7:      else
8:          return false
9:      end if
10: end function
```

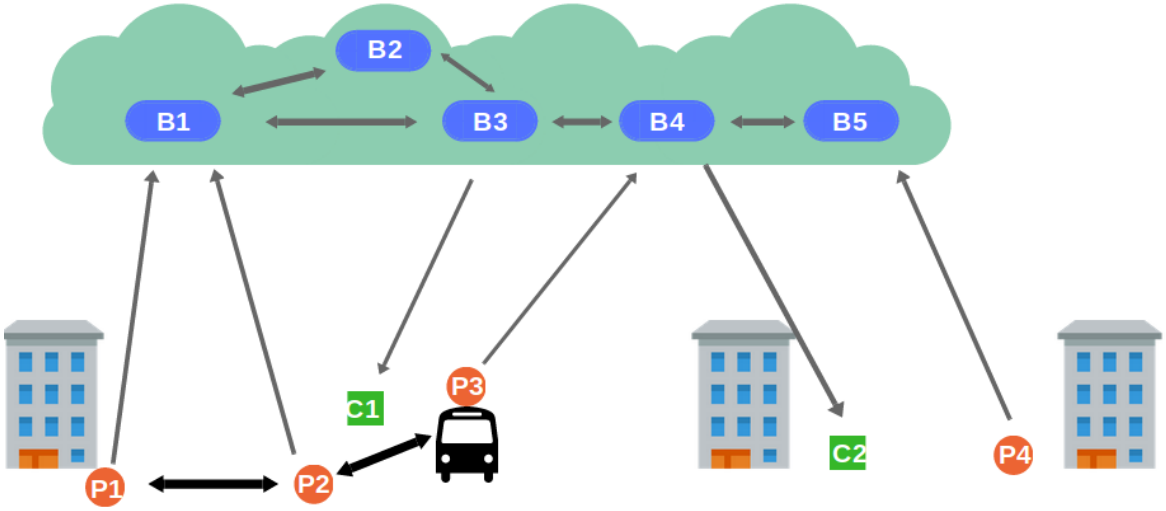


Figure 4. Collaborative architecture

5.3. Collaborative architecture

To reduce the burden of having brokers responsible for managing producers' location, we have designed a third architecture that minimizes the broker role by placing location verification on the producer side. This architecture relies on a traditional cloud architecture. Producers can be connected to any border broker, which may be different from the one their close neighbors are connected to. As depicted in Figure 4, this architecture incorporates short-range connections between nearby producers so they can communicate with each other and collaboratively decide the location they tag their notifications with. Every producer is equipped with a local broker that is in charge of handling the notification and the location provided by the producer. Then, it communicates with local brokers of nearby producers. Finally, it sends the notification to a border broker in the cloud with the collaboratively decided location. In our previous work [12], we proposed the use of a consensus strategy as the mechanism for collaborative location decision. However, due to the high mobility of the producers, which arrive at or leave an area at any time, the possibility of forming a stable group of nodes that satisfies the requirements to take part in a consensus agreement is low. As a result, we have decided to employ an alternative collaborative strategy: neighbor polling.

Every time a publication is created, the local broker polls nearby producers by sending a short-range broadcast message. Thus, collaboration is restricted to nodes in the same physical area. The local broker waits for a certain time, computes the location values received and includes in the publication the decided value. The location value proposed locally is also included in the poll. The decided location is the most repeated location in the poll, as long as at least one neighbor answer was received and there is no tie between the most voted locations. This requirement can be written as:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

1. $RL > 1$
2. $\exists! b \in R, b = max(R)$

where RL is the list of received locations plus the one provided by the local broker, which consists of entries formed by two triples: $e = \{(\text{location}, \text{value}, X), (\text{numberOfReplies}, \text{value}, Z)\}$.
R is the list that includes the number of poll replies for every location. That is:
$R \leftarrow \{Y | \{(\text{location}, \text{value}, X), (\text{numberOfReplies}, *, Y)\} \in RL\}$.
Publications issued to the broker include an indication of whether the location has been collaboratively decided, in the form of the triple: $(\text{location}, \text{collaborativelyDecided}, \text{X})$, where X is a boolean variable whose value is *true* if the collaboration has been successful. In case no poll replies are received, or there is a tie between the most voted locations in the poll, the result is undecided. Then, the location proposed locally is included in the notification together with an indication of unverified location — i.e, a triple where X takes the value $false$. Algorithm 8 describes the location verification procedure for this strategy.

Algorithm 8 : at producer P, prepare extra location information (Collaborative architecture)

```
1:  function PrepareExtraLocationInformation
2:      RL, R ← ComputePollReplies()
3:      if sum(R) > 1 ∧ (∃!b ∈ R, b = max(R)) then
4:          collaborativeLocation ← X|{(location, value, X), (numberOfReplies, value, max(R))} ∈ RL
5:          lc ← {(location, value, collaborativeLocation), (location, collaborativelyDecided, true)}
6:      else
7:          lc ← {(location, value, locallyProposedLocation), (location, collaborativelyDecided, false)}
8:      end if
9:      return lc
10: end function
```

Notifications with a location not collaboratively verified are not certified by the broker. In this architecture, the role of the broker is reduced to checking the collaborative verification indication and certifying the notifications accordingly (Algorithm 9). It is responsible for providing location certifications but does not decide when to provide them.
The local broker is assumed to send the location agreed in the poll truthfully. However, when polled by a neighbor, it answers with the location value proposed locally, which may not be true.

Algorithm 9 : at broker B, verify location (Collaborative architecture)

```
1: function VerifyLocation(Notification n)
2:     locCertified ← {Y|(location, collaborativelyDecided, Y) ∈ n}
3:     remove triple (location, collaborativelyDecided, *) from n
4:     return locCertified
5: end function
```

6. Simulation

To evaluate our architectures, we rely on network simulation. We simulate the noise measurement scenario described in Section 3. To do so, we employ realistic pedestrian traces based on a real urban map. We have chosen the city center of Vigo (Spain), an area of 1.46 km^2 where most of the streets are either pedestrian-only or include wide sidewalks, which makes it appropriate for pedestrian simulation. We have generated pedestrian traces for 100 pedestrians using Bonnmotion [21] mobility scenario generation tool and a map of the selected area exported from OpenStreetMap [22]. The traces are generated according to MSLAW [23], a map-based statistical mobility model that extends the Self-similar Least-Action Walk (SLAW) model by including geographic restrictions. As a result, it targets realistic mobility, considering different mobility metrics such as pauses and inter-contact time. We have generated five different mobility traces. Then, we have fed those traces to ns-3 network simulator [24], where we have implemented our proposed architectures. We have chosen ad hoc WiFi as the transmission technology for short-range communication since its transmission range ($\approx$ 100m)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

Table 1. Shared simulation parameters

Short-range communication	WiFi ad hoc mode Connection type: Direct Connection pattern: Random
Cloud communication	LTE
Scenario	Number of producers: 100 Simulation duration: 1h Simulation area: 1.46km^2
Mobility	Model: MSLAW Speed: 0.9 – 1.5 m/s Pause time: 10 – 50 s
Parameters	Notification interval: 1 minute

makes it suitable for urban scenarios. Moreover, we have modeled cloud communication through LTE connections.

To implement our verification strategies, our area is divided into smaller cells. Each of them is assigned a cell identifier, which is the location value producers use as location attribute in their publications. We assume producers are always aware of what is their position and the identifier of the cell they are in.

Our proposed architectures aim at correctly identifying the veracity of the location claims included in publications at the minimum cost. This cost is considered in terms of required infrastructure and consequences in publication delivery. Thus, the assessment is focused on the appropriate location certifications and at the ability of the system to handle producers' mobility. In that respect, we have carried out simulations where producers choose to provide a fake location according to a certain probability (P_f), which follows a uniform distribution.

For the sake of simplicity and without impairing correctness, we have not included consumers in our simulation scenarios. Since we aim to assess publication creation in producers and publication certification in brokers, broker-consumer communication is out of the scope of this work.

The publication interval has been set to 1 minute, which leads to frequent sampling times without excessively draining mobile devices' battery.

Table 1 recaps the main parameters of the simulation that are common to the three architectures. This section continues with the description of the implementation details specific to each architecture.

6.1. Fixed brokers architecture

This simulation scenario includes mobile producers and fixed brokers, all of them equipped with ad hoc WiFi. The area is divided in a grid of square of cells of ≈ 200m x 200 m. There is a fixed broker in the middle of every cell. A number identifier is assigned both the broker and the cell. When they issue a publication, producers include as a location tag the cell identifier that corresponds to their location coordinates. Brokers provide a location certification to every publication they receive whose cell identifier corresponds to their own.

Producers establish a connection with a broker from whom they have received a broker beacon when they are not previously connected to any other. When connected, they send their periodical publications to their broker. If not connected, notifications are queued in the producer.

Our work focuses on designing a communication strategy that could allow mobile producers to stay connected most of the time to their nearest broker despite their continuous movement. Thus, the number of publications lost and not transmitted are minimized while the number of publications whose location claim is correctly assessed is maximized. With this goal in mind, extensive simulations are performed testing different time values for broker beacon interval and maximum connection time. The latter is the time after which producers consider themselves disconnected from a certain broker if they have ceased to receive beacons. It was found that publication delivery rate improves when the maximum connection time matches the broker beacon interval. Moreover, as for pedestrian mobility, this time is best set to 2s.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

6.2. Assigned brokers architecture

This simulation scenario includes mobile producers that communicate with brokers in the cloud using LTE technology. Producers act as LTE user equipment (UE) and connect to a base station (eNB) that covers the whole simulation area. Producers are also equipped with ad hoc WiFi, which allows them to interact with other producers nearby.
There is a broker assigned to every one of the cells in the area, which is in charge of managing the publications issued by producers there. Producers are aware of the cell they are in and include the cell identifier in the publications they issue. Moreover, they also include a list of their neighbor producers, which they have detected in their surroundings by using periodic ad hoc WiFi beaconing. Producers are assumed to always provide their true neighbor list, even when they lie about their location.
The size of the cells has been decided as a function of ad hoc WiFi transmission range ($\approx$ 100 m). The size in the first architecture ($\approx$ 200m x 200 m) allows fixed brokers to cover the whole area from their position in the middle. However, in this scenario, areas need to be smaller ($\approx$ 100 x 100 m) so that producers are able to communicate with most of their peers in the area. As a result, producers are also more likely to receive beacons from their neighbor areas.
If periodic beaconing between producers is more frequent than publications, producers can detect changes in their surroundings faster than the broker. This may result in incorrect publication certification. As a result, we have set the publication interval much lower than in the previous simulation, as low as the beaconing interval. To have a fair comparison with the first architecture, notifications shown in the results section have been sampled every minute.
The simulation includes a warm-up period where notifications are not accounted for and producers provide only true locations. Thus, false location claims are always dealt with in a steady state. This period has been set to 30s. Since the beaconing and notification intervals have been set to 2 seconds each, this is enough for sufficient notifications to be processed.

6.3. Collaborative architecture

The simulation setup is similar to the one in the assigned brokers architecture: producers act as LTE user equipment (UE) connected to a base station (eNB) that covers the simulation area. However, the grid distribution adopted is the same as in the fixed brokers architecture, where the area is divided in a grid of 200 x 200 m cells. This cell size offers an appropriate balance between location precision and reduced likelihood of receiving too many poll replies from neighbor areas. Producers can be connected to any access broker regardless of their position.
Moreover, producers communicate with their neighbors using ad hoc WiFI when polling or answering a poll. There is no beaconing in this architecture.
In collaborative location assessment, producers are in charge of verifying location claims without brokers being involved. Given that local producers independently decide to provide a false location claim, this strategy makes it unlikely that a notification with a false claim is certified. For this to happen, at least two neighbor producers have to provide the same fake claim. Moreover, the rest of the poll participants (if any) have to provide location claims that either do not coincide with each other or do not add up to more than the fake location claims.
Waiting time for poll answers has been set to 2s, enough time for producers to receive their peers' answers without too much delay in the publication process.

7. Evaluation

To assess our architectures, we examine the number of publications, how they are transmitted and how they are evaluated for certification. Thus, we evaluate the number of publications that are certified and the cost of such certification in every one of the architectures, in terms of lost and incorrectly certified and uncertified publications.
For every architecture, we have run simulations where producers never provide fake location claims

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

($P_f = 0$), and simulations where they may provide a fake claim ($P_f = 0.3$). Simulation results in Tables 2 and 3 show the average results obtained after simulating every architecture with 5 different trace files. The tables include a row for every verification strategy proposed in Section 5: one for the fixed brokers architecture, two for the assigned brokers architecture and one for the collaborative architecture. The tables are divided into three parts. The first one (a) presents the flow of notifications in the producer and through the network. The second (b) and third (c) parts depict the notifications that are delivered to an access broker and how they are processed. These subtables display the treatment of the notifications with true and false location claims, respectively. For consistency, the tables include the same columns for the different verification strategies and probabilities of fake location claims (P_f). However, it should be noted that some of the columns are only relevant to a certain strategy or probability.

Table 2. Simulation statistics $P_f = 0$

(a) Producer and network flow

	Published			Sent		Queued	Lost	Delivered
	Total	True loc	False loc	True loc	False loc			
Fixed	5900	5,900	0	5,900	0	0.2	62.6	5,837.2
Assigned-CLS	5900	5,900	0	5,900	0	0	0	5,900
Assigned-NLS	5900	5,900	0	5,900	0	0	0	5,900
Collaborative	5900	5,900	0	5,573.4	326.6	0	0	5,900

(b) Delivered notifications with a true location claim

	Delivered				
	True location				
	Total	Cert	%	Uncert	%
Fixed	5,837.2	5,275.8	90.38	561.4	9.62
Assigned-CLS	5,900	4,005.4	67.89	1,894.6	32.11
Assigned-NLS	5,900	631.8	10.71	5,268.2	89.29
Collaborative	5,573.4	4,073	73.08	1,500.4	26.92

(c) Delivered notifications with a false location claim

	Delivered				
	False location				
	Total	Cert	%	Uncert	%
Fixed	0	0	0	0	0
Assigned-CLS	0	0	0	0	0
Assigned-NLS	0	0	0	0	0
Collaborative	326.6	326.6	100	0	0

The first subtable (a) starts with the number of notifications created by the producers (i.e.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

Published) and describes the distribution of true and false location claims between those. In the collaborative architecture, the notifications are processed by the local broker before being sent into the network. As a result of the collaborative verification process, their location claims may change and therefore, the distribution of true and false locations claims can be different. This distribution is reflected in the *Sent* column. Sent notifications may not be delivered to the access broker, they may remain in the producers' queue at the end of the simulation (*Queued*) or they may simply not reach the broker due to network issues (*Lost*). The second subtable (b) displays the number of notifications delivered to an access broker that include a true location claim and how many of them are certified and uncertified by the broker. The rates of certified and uncertified delivered notifications with a true location claim are also expressed as a percentage. The third subtable (c) is identical to the second one but considering delivered notifications with a false location claim instead.

Table 3. Simulation statistics $P_f = 0.3$

(a) Producer and network flow

	Published			Sent		Queued	Lost	Delivered
	Total	True loc	False loc	True loc	False loc			
Fixed	5900	4,106.8	1,793.2	4,106.8	1,793.2	0.2	62.6	5,837.2
Assigned-CLS	5900	4,088.2	1,811.8	4,088.2	1,811.8	0	0	5,900
Assigned-NLS	5900	4,088.2	1,811.8	4,088.2	1,811.8	0	0	5,900
Collaborative	5900	4,137	1,763	4,534	1,366	0	0	5,900

(b) Delivered notifications with a true location claim

	Delivered				
	True location				
	Total	Cert	%	Uncert	%
Fixed	4,062.2	3,670	90.35	392.2	9.65
Assigned-CLS	4,088.2	2,542.6	62.19	1,545.6	37.81
Assigned-NLS	4,088.2	520	12.72	3,568.2	87.28
Collaborative	4,534	3,220	71.02	1,314	28.98

(c) Delivered notifications with a false location claim

	Delivered				
	False location				
	Total	Cert	%	Uncert	%
Fixed	1,775	0	0	1,775	100
Assigned-CLS	1,811.8	238.4	13.16	1,573.4	86.84
Assigned-NLS	1,811.8	2.2	0.12	1,809.65	99.88
Collaborative	1,366	366	26.79	1,000	73.21

The number of published notifications is the same for all the architectures while, due to their

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

random generation, the number of publications with a false location claim slightly differs. The number of publications sent only differs with the number of published ones in the collaborative architecture, where the location is the result of the neighbor poll and not directly the value decided by the producer itself. Publications are only queued or lost in the fixed brokers architecture, as explained in 7.1.1. In the other two architectures, producers communicate with their brokers using LTE technology and, since LTE coverage of the area is total, producers are always able to send their publications. Therefore, there are neither queued nor lost notifications.
All the architectures present solid results in detecting false location claims. However, their simulation outcomes differ. In this section, we examine the statistics of each architecture simulation in detail.

7.1. Fixed brokers architecture

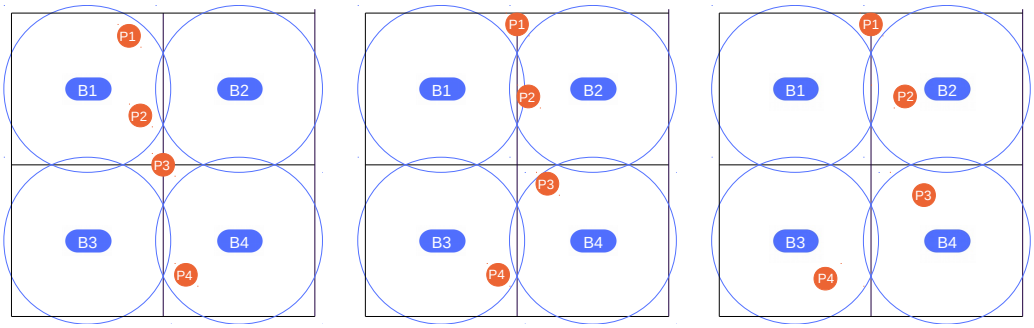
In this architecture, the average rate of messages lost is slightly over 1% (Tables 2.a and 3.a, $62.6/5900 = 1.061\%$) while there are practically no queued messages. Thus, in the worst case, at least 98% of all the publications are received at an access broker and transmitted. Results are also good for the correct certification rate. Every message with a false location is correctly identified and therefore none of them is certified (Tables 2.c and 3.c) . In other words, 100% of messages with false location are successfully assessed. The rate of messages incorrectly uncertified does not reach 10% (Tables 2.b and 3.b). As a consequence, more than 90% of messages with true location are correctly identified. In general terms, we can conclude that the system reasonably meets its target of delivering publications to the broker and correctly providing location certifications.

7.1.1. Publication and certification issues

Due to the producer mobility, it is difficult to completely solve publication loss and certification issues. There exist queued publications that never leave their source producer during the simulation time and therefore are not delivered to any broker. These publications may have been transmitted if the simulations were longer. Lost publications are those that have been sent by a producer but they have not reached an access broker. In this architecture, the certification issue affects mostly publications with a true location claim that are not certified due to their mobility.
Figure 5 illustrates situations where publications are lost or incorrectly uncertified. It represents a grid divided into four different areas, with a broker placed on the center of each one. The areas have been designed to maximize area coverage with the minimum number of brokers while avoiding overlapping areas. The circles around the brokers represent the area covered by their signal. Each of the subfigures represents a different temporal instant. Thus, the whole Figure 5 depicts a temporal sequence where producers are moving through the area. We assume that brokers send beacons at t and $t + 2$ and that producers issue a periodic publication at $t + 1$.
At t, Producers 1 and 2 being in the same area as Broker 1, they receive a beacon from this broker and establish a connection with it. Producer 4 connects in turn to Broker 4. Producer 3, however, is in an area without coverage so it does not receive any beacon and considers itself as being disconnected.
At $t + 1$, all the producers have moved. However, they have not received any new broker beacon, and neither have they missed it since the maximum connection time (2 s) has not expired yet. As a result, they will consider their previous situation when delivering a publication. Although Producer 1 considers that it is still connected to Broker 1, it is in an area without coverage and, as a consequence, its publication is lost. The same happens to the event notification delivered by Producer 4 since it is issued to Broker 4, which is not anymore in the transmission range. Producer 2 has also changed area but, since it is on an overlapping edge of the new area, it is still able to communicate with Broker 1. However, its publication includes a location claim of area 2 and, therefore, it cannot be certified. Since the notification includes the actual area where the producers is at the publication time but it is processed by a broker in charge of a different area, Producer 2's publication becomes an uncertified publication with true location. Producer 3 has entered Broker 4's area but, since it has not established

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104



(a) Producers at time t **(b)** Producers at time $t + 1s$ **(c)** Producers at time $t + 2s$

Figure 5. Example of area grid and producers movement

any connection, it saves the publication in its internal queue.
At $t + 2$, most of the producers have moved again, except for Producer 1. In doing so, Producer 1 does not receive any beacon and realizes that its connection with Broker 1 is broken. Producers 2, 3 and 4 receive beacons from their nearest brokers (Brokers 2, 4 and 3, respectively) and establish a connection with them.
The examples above describe what may happen when nodes are on an area edge. If they leave the area, it takes some time for them to realize that they cannot reach anymore the broker they were connected to. Consequently, they may send a publication that never reaches the broker and is then lost (e.g., Producer 1 at $t + 1$). Publications may also become lost due to collisions on the WiFi channel. It is also possible that nodes move to an overlapping area that corresponds to a new broker. In this case, they will send publications to their old broker that include their new location (e.g., Producer 2 at $t + 1$). This is one of the situations where publications with true location are not certified. The other possible scenario takes place when nodes that have lost their connection for some time find another broker. Then, they issue their queued publications to the new broker.

7.2. Assigned brokers architecture

Since the verification strategy relies only on mobile producers, simulation results cannot meet the ones obtained in the fixed-brokers architecture. Neighbor changes are quick, even more with smaller areas, and therefore, it is difficult for the producers to be always aware of who is nearby. The percentage of notifications with a false location claim that are certified is low for both of the verification strategies (Tables 2.c and 3.c), but is zero in NLS because false-location notifications do not include any neighbor that is registered in the broker. Publications that are incorrectly certified in CLS satisfy the conditions required as they did not include any neighbor unknown to the broker nor fail to include a neighbor known to the broker. They are publications that include an empty neighbor list sent to a broker that do not have any producer registered in the area. That is to say; they are labeled with a location claim that belongs to an empty area.
Even though the systems correctly identifies most of the notifications with a false location claim, it does not work that well at recognizing true claims (Tables 2.b and 3.b). The performance is especially poor in NLS. Node density and the number of cells in the grid play a role in this situation. Due to the cell distribution, there are many more areas than producers. As a result, it is not unlikely that a producer is alone in this cell. Although there is a risk in certifying lonely producers as explained in the previous paragraph, it pays off in terms of general results in a low-density scenario like this.
Nodes providing false claims also interfere in the verification of the publications issued by their neighbors. A publication including a neighbor that has not been registered in the broker results in uncertification in CLS. This may happen when a producer includes an actual neighbor whose

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

publication provides a false claim. Also, when a publication is wrongly certified, the producer is included in the list of producers registered with the broker assigned to that area. For a certain time, this will prevent true notifications from being certified since the broker will not certify publications that do not include this producer, which is not known by any of the actual producers in the area.
The verification strategies have different strengths and therefore can be applied to different scenarios. CLS is better suited for scenarios where the overall performance of the system is considered. In situations where it is of paramount importance to avoid false location claims, NLS should be employed. It is successful as long as the assumption of producers sending their true neighbor list is maintained.

7.3. Collaborative architecture

Since location values in publications may change after the collaborative assessment, the analysis of this architecture is more complex. Tables 2 and 3 show first a distribution of true and false location claims before (*Published*) and after (*Sent*) collaborative verification. Then, they present the same distribution once the publications are delivered to the broker (*Delivered*). In the first two architectures, location values in notifications do not change during the verification process. Therefore, this classification is sufficient to understand the operation of the verification strategies. However, in the collaborative case, it is necessary to study how location claims are modified to assess the outcome of the verification strategy.
With that aim, Tables 4 and 5 provide a more detailed study of the results in Tables 2 and 3. First, Table 4 presents the number of delivered notifications and the rates of certification. Then, Table 5 takes a closer look at the veracity of the location claims of certified and uncertified notifications and classifies them according to whether they have been modified in the verification process.

Table 4. Certification statistics for delivered notifications in the collaborative architecture simulation

Delivered					
	Total	Certified		Uncertified	
P_f		Total	%	Total	%
0	5,900	4,399.6	74.57	1,500.4	25.43
0.3	5,900	3,586	60.78	2,314	39.22

The location claims in certified notifications have always been collaboratively decided. As a result, the original claim inserted by the producer may have been replaced with a different value, which may have altered the veracity of the claim. Consequently, a true location claim can either maintain its value (*Remain true*) or have had it changed for a false one (*True to false*). Likewise, a false location claim may still be false (*Remain false*) or may have been corrected to a valid value (*False to true*). *Remain false* includes both unchanged claims and claims that have been substituted by a different value, also incorrect. On the other hand, uncertified notifications contain always the original location claims proposed by the producers, i.e., they maintain their initial veracity.
The results show how most of the notifications published are certified (Table 4), which means the location they include has been collaboratively assessed. Out of these, most of the notifications included originally a true location that was maintained after the collaborative evaluation or initially presented a false claim that was corrected (Table 5.a). As a matter of fact, a third of the total publications with false claims were corrected to their true location (598 corrected, 1165 remain false ($165 + 1000$)) while only about 5% of the true location claims were modified (201 vs 3936 unchanged ($2622 + 1314$)).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

Table 5. Detailed statistics for delivered notifications in the collaborative architecture simulation

(a) Certified notifications

Certified									
P_f	Total	Remain true		False to true		Remain false		True to false	
		Total	%	Total	%	Total	%	Total	%
0	4,399.6	4,073	92.58	0	0	0	0	326.6	7.42
0.3	3,586	2,622	73.12	598	16.68	165	4.6	201	5.61

(b) Uncertified notifications

Uncertified					
P_f	Total	Remain true		Remain false	
		Total	%	Total	%
0	1,500.4	1,500.4	100	0	0
0.3	2,314	1,314	56.78	1,000	43.22

Every modified claim is certified since all of them are decided through collaborative assessment. Even though there are notifications that remain or become false after the assessment (Table 5.a), it should be noted that their locations always correspond to neighbor areas. This way, originally false location claims that are certified and that are not corrected to their true location, are at least changed to an adjacent location claim. This also happens with originally true location claims, which are modified to a false value when their producers are on an area limit and therefore, receive multiple poll replies from a neighbor area. As a result, every false certified notification comes from a producer standing on the edge of an area and consequently close to the neighbor location included in their notification. Consequently, their claims are not useless but give a valid clue about the actual producer location.
Uncertified notifications are those whose collaborative assessment is declared undecided and then include the original location proposed by the producer (Table 5.b). Two factors contribute to the number of uncertified notifications. The first one is the number of tied polls. When producers receive poll replies from different areas and there is a draw between the results, the notification is uncertified. The percentage of true uncertified notifications increases in the simulation with false notification claims since lying producers may create draws in polls where all the producers are in the same area. Polls with only two participants (i.e have received only one reply) are especially sensitive to this. In this case, if the two location claims proposed coincide, the poll result is certified whereas a poll with two different location claims results in uncertification. There is a significant number of minimal polls in the simulation. This is caused by the second factor involved in uncertification: low density of devices. Although 100 nodes is realistic density for a quiet urban area, it may be slightly low for a collaborative scheme like this. In conclusion, notifications from producers that do not receive replies are always uncertified and notifications from producers that receive only one reply are sensitive to be also uncertified.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

Table 6. Differences between architectures

	Fixed Brokers	Assigned Brokers CLS	Assigned Brokers NLS	Collaborative
Cell size	200 x 200 m	100 x 100 m		200 x 200 m
Dedicated infrastructure	Yes	No		No
Intermittent broker connection	Yes	No		No
Periodic beaconing	Yes	Yes		No
Broker role	Principal	Intermediate		Minimal
Sensibility to producers' density	None	High		High
Infrastructure	Physical	Cloud		None
Reliability	High	High	Intermediate	Intermediate

7.4. Discussion

Although the three architectures can be successfully employed for location verification in DEBS, each of them is more suitable for a different scenario. In order to paint a clear picture of the simulation setup and enable a fair result comparison, the main differences between the architectures are outlined in Table 6. The table starts by presenting the different cell sizes employed in the architectures. Then, it indicates which of them require dedicated infrastructure and periodic beacons. Moreover, the table details the different levels of broker involvement and neighbor dependency the architectures entail.

The fixed brokers architecture is always the most reliable option and provides the best results, despite losses. As a result, it is advisable in scenarios where dependability is crucial and a high investment in infrastructure is possible. Moreover, this architecture does not rely on the existence of neighbor producers and works well with low producer density.

At the opposite end of the spectrum, the collaborative architecture fits best in flexible scenarios, where no resources are required to be allocated for broker assignment. Although polling is costly, it is not as much as periodic beacons in the other architectures. However, this architecture certifies the most publications with false location claims (Tables 2.c and 3.c). It should be noted that the location of this publications has indeed been corrected but to an area adjacent to the correct one. As a result, the information is more useful than the original false location claim. Nevertheless, this architecture is less appropriate in scenarios where precision is essential.

The first verification strategy of the assigned brokers architecture (CLS) achieves moderately better results by introducing the burden of the broker allocation. NLS is the best alternative when infrastructure deployment is not possible and the system is less tolerant to incorrectly certified claims. If the system has a tolerance, CLS and the collaborative architecture present a better percentage of correctly certified notifications since they work better with true location claims.

8. Related Work

Due to the rise of participatory sensing schemes, it has become necessary to develop solutions to verify the quality and correctness of the contributed data. Data verification strategies can be general, to cover any kind of information, or more specific, targeted at a certain type, such as location.

An example of general data verification is SHIELD [25], a centralized system based on evidence handling and data mining to identify and filter out erroneous user contributions. However, the requirement for a bootstrapping phase makes data mining unsuitable for a dynamic distributed scenario like ours.

Moving away from classification and training, Luo and Zeynalvand [26] propose a cross validation

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

framework that recruits crowdworkers to act as data verifiers. Their role is not to duplicate the sensing task but to assess the credibility of the data gathered. This approach is also centralized and, unlike ours, relies on verifiers with no direct knowledge about the information.

The reason why location verification has attracted significant research attention is twofold. On the one hand, location information plays an important role in contextualizing contributed information. On the other hand, location-based services require users to provide a valid location value in order to gain access. The proposals reviewed next are targeted at these aims.

Lederer et al. [27] introduce a certification approach at network level where intermediate nodes tag data through the network with belief ratings based on observed past traffic patterns and source location claims. Relying on previous knowledge information about routing paths is a valid strategy for distributed systems where nodes do not move often and, therefore, paths are not likely to change. As a result, this strategy is not appropriate for a highly dynamic scenario like the one we consider.

Most location verification mechanisms rely on the collaboration of neighbor devices. Although not specifically labeled as fog architectures, these strategies consist of cloud systems that leverage neighbor information. Saracino et al. [28] propose a reputation-based system where some of the devices in each area act as hotspots to communicate with their peers in WiFi range. A centralized platform is in charge of receiving the lists of devices that can be reached and update reputation calculations accordingly. ILR [29] aims at detecting false location claims in a centralized crowdsensing system where data providers are assigned sensing tasks. When submitting the required data, they also include a Bluetooth scan of their surroundings. False location claims are identified once the tasks are received, by comparing Bluetooth scans and also through image processing when the task includes picture taking.

Unlike ILR, other mechanisms are not post processing, but designed to be used at runtime. LINK [30] is targeted at location verification for third-party location-based services. It relies on a Location Certification Authority (LCA) that receives information on location claimers and their Bluetooth neighbors. Then, it issues a claim decision to the service the claimer wants to authenticate for.

APPLAUS [31], STAMP [32], PROPS [33] and SPARSE [34] share LINK's aim but include extra features to deal with privacy and colluding neighbors. Instead of exchanging real identifiers, devices in APPLAUS [31] use pseudonyms and change them periodically. PROPS [33] uses encryption to preserve the identity of the claimer and its neighbors. Regarding collusion prevention, STAMP [32] pays attention to transaction history between two users to detect if they appear to have always the same neighbors. In SPARSE [34] the central entity of the system decides which of the neighbors should act as location witnesses, avoiding that location claimers can select them themselves.

All these location verification schemes that involve neighbor collaboration are centralized and aimed at sporadic location verification. Therefore, they have not been designed to handle continuous sensing in a very distributed setting.

9. Conclusion and Future Work

In this paper, we have demonstrated the potential of fog-based architectures in the implementation of participatory urban DEBS with location verification support. We have proposed three different architectures: the fixed brokers architecture, the assigned brokers architecture and the collaborative architecture. We have proven them to successfully deal with location verification in realistic urban scenarios and we have discussed which situations would best benefit from each one of them.

Due to the increasing adoption of mobile sensor-enabled devices, the number of services oriented at them or exploiting their possibilities will continue to grow over the next few years. As a result, the demand for location verification mechanisms targeted at them will also escalate. Our work is aimed at verifying producer location but it can easily be adapted for consumers. Thus, event subscriptions could be restricted to consumers in a particular area. This first perspective of our work is related to our previous work in multiscale DEBS [35].

Proximity-based collaboration between peers can be extended to verify additional context-related

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

information. However, it will increase the complexity of event processing in the broker. Moreover, other context-related information different from temporal and spatial details are mostly application-dependent. Therefore, these collaborative solutions, albeit useful, would be less general and restricted to specific scenarios. As a second perspective, in order to extend the functionality of the system, this work could benefit from analyzing group mobility [36].

Author Contributions: Conceptualization, F.C.-J., D.C. and S.C.; Methodology, F.C.-J., D.C., R.P.D.-R. and A.F.-V.; Software and Validation, F.C.-J.; Formal Analysis, F.C.-J., A. F.-V. and R.P.D.-R.; Writing—Original Draft Preparation, F.C-J., R.P.D.-R. and A.F.-V.; Writing—Review & Editing, F.C-J., R.P.D.-R., A.F.-V., S.C. and D.C.; Funding Acquisition, R.P.D.-R. and A.F.-V. All of the authors approved the final version of the manuscript.

Funding: This work is funded by: the European Regional Development Fund (ERDF) and the Galician Regional Government under agreement for funding the Atlantic Research Center for Information and Communication Technologies (AtlantTIC), the Spanish Ministry of Economy and Competitiveness under the National Science Program (TEC2014-54335-C4-3-R and TEC2017-84197-C4-2-R) and a predoctoral grant financed by the Galician Regional Government (Consellería de Cultura, Educación e Ordenación Universitaria) and the European Social Fund.

Conflicts of Interest: The authors declare no conflict of interest

1. Mone, G. The new smart cities. *Communications of the ACM* **2015**, *58*, 20–21.
2. Eugster, P.; Felber, P.; Guerraoui, R.; Kermarrec, A.M. The Many Faces of Publish/Subscribe. *ACM Computing Surveys* **2003**, *35(2)*.
3. Antonic, A.; Roankovic, K.; Marjanovic, M.; Pripuic, K.; others. A mobile crowdsensing ecosystem enabled by a cloud-based publish/subscribe middleware. FiCloud 2014. IEEE, 2014, pp. 107–114.
4. Cugola, G.; Margara, A. Processing flows of information: From data stream to complex event processing. *ACM Computing Surveys (CSUR)* **2012**, *44*, 15.
5. Bellavista, P.; Corradi, A.; Fanelli, M.; Foschini, L. A survey of context data distribution for mobile ubiquitous systems. *ACM Computing Surveys (CSUR)* **2012**, *44*, 24.
6. Burke, J.; Estrin, D.; Hansen, M.; Parker, A.; Ramanathan, N.; Reddy, S.; Srivastava, M. Participatory sensing. *Center for Embedded Network Sensing* **2006**.
7. Guo, B.; Yu, Z.; Zhou, X.; Zhang, D. From participatory sensing to mobile crowd sensing. PERCOM 2014. IEEE, 2014, pp. 593–598.
8. Castro-Jul, F.; Díaz-Redondo, R.P.; Fernández-Vilas, A. Collaboratively assessing urban alerts in ad hoc participatory sensing. *Computer Networks* **2018**, *131*, 129–143.
9. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. MCC'12. ACM, 2012, pp. 13–16. doi:10.1145/2342509.2342513.
10. Perera, C.; Qin, Y.; Estrella, J.C.; Reiff-Marganiec, S.; Vasilakos, A.V. Fog computing for sustainable smart cities: A survey. *ACM Computing Surveys (CSUR)* **2017**, *50*, 32.
11. Lim, L.; Marie, P.; Conan, D.; Chabridon, S.; Desprats, T.; Manzoor, A. Enhancing Context Data Distribution for the Internet of Things using QoC-awareness and Attribute-based Access Control. *Annals of Telecommunications* **2016**, *71(3)*, 121–132. doi:10.1007/s12243-015-0480-9.
12. Castro-Jul, F.; Conan, D.; Chabridon, S.; Díaz-Redondo, R.P.; Fernández-Vilas, A.; Taconet, C. Combining Fog Architectures and Distributed Event-Based Systems for Mobile Sensor Location Certification. International Conference on Ubiquitous Computing and Ambient Intelligence. Springer, 2017, pp. 27–33.
13. Ganti, R.K.; Ye, F.; Lei, H. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine* **2011**, *49*.
14. Antonić, A.; Marjanović, M.; Pripužić, K.; Žarko, I.P. A mobile crowd sensing ecosystem enabled by CUPUS: Cloud-based publish/subscribe middleware for the Internet of Things. *Future Generation Computer Systems* **2016**, *56*, 607–622.
15. Dang, T.D.; Hoang, D. A data protection model for fog computing. Second International Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2017, pp. 32–38.
16. Yannuzzi, M.; Milito, R.; Serral-Gracià, R.; Montero, D.; Nemirovsky, M. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. CAMAD 2014. IEEE, 2014, pp. 325–329.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 November 2018 doi:10.20944/preprints201811.0407.v1

Peer-reviewed version available at Sensors 2018, 19, 104; doi:10.3390/s19010104

17. Hong, K.; Lillethun, D.; Ramachandran, U.; Ottenwälder, B.; Koldehofe, B. Mobile Fog: A Programming Model for Large-scale Applications on the Internet of Things. ACM SIGCOMM, 2013, MCC '13, pp. 15–20. doi:10.1145/2491266.2491270.
18. D'Hondt, E.; Stevens, M.; Jacobs, A. Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring. *Pervasive and Mobile Computing* **2013**, *9*, 681–694.
19. Etzion, O.; Niblett, P.; Luckham, D.C. *Event processing in action*; Manning Greenwich, 2011.
20. Fiege, L.; Gartner, F.C.; Kasten, O.; Zeidler, A. Supporting mobility in content-based publish/subscribe middleware. ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing. Springer, 2003, pp. 103–122.
21. Aschenbruck, N.; Ernst, R.; Gerhards-Padilla, E.; Schwamborn, M. BonnMotion: a mobility scenario generation and analysis tool. Proceedings of the 3rd international ICST conference on simulation tools and techniques. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010, p. 51.
22. OpenStreetMap. http://wiki.openstreetmap.org/, 2015.
23. Schwamborn, M.; Aschenbruck, N. Introducing geographic restrictions to the slaw human mobility model. 21st International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS). IEEE, 2013, pp. 264–272.
24. NS - 3 network simulator. https://www.nsnam.org/, 2015.
25. Gisdakis, S.; Giannetsos, T.; Papadimitratos, P. SHIELD: A data verification framework for participatory sensing systems. Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks. ACM, 2015, p. 16.
26. Luo, T.; Zeynalvand, L. Reshaping Mobile Crowd Sensing using Cross Validation to Improve Data Credibility. Global Communications Conference (GLOBECOM). IEEE, 2017, pp. 1–7.
27. Lederer, S.; Gao, J.; Sion, R. Collaborative Location Certification for Ad-Hoc Sensor Networks. Sarnoff Symposium. IEEE, 2008, pp. 1–6.
28. Saracino, A.; Restuccia, F.; Martinelli, F. Practical Location Validation in Participatory Sensing Through Mobile WiFi Hotspots. 17th International Conference On Trust, Security And Privacy In Computing And Communications/ 12th International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018, pp. 596–607.
29. Talasila, M.; Curtmola, R.; Borcea, C. Improving location reliability in crowd sensed data with minimal efforts. 6th Joint IFIP Wireless and Mobile Networking Conference (WMNC). IEEE, 2013, pp. 1–8.
30. Talasila, M.; Curtmola, R.; Borcea, C. Link: Location verification through immediate neighbors knowledge. International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services. Springer, 2010, pp. 210–223.
31. Zhu, Z.; Cao, G. Toward privacy preserving and collusion resistance in a location proof updating system. *IEEE Transactions on Mobile Computing* **2013**, *12 (1)*, 51–64.
32. Wang, X.; Pande, A.; Zhu, J.; Mohapatra, P. STAMP: enabling privacy-preserving location proofs for mobile users. *IEEE/ACM transactions on networking* **2016**, *24*, 3276–3289.
33. Gambs, S.; Killijian, M.O.; Roy, M.; Traoré, M. PROPS: A privacy-preserving location proof system. 33rd International Symposium on Reliable Distributed Systems (SRDS). IEEE, 2014, pp. 1–10.
34. Nosouhi, M.R.; Yu, S.; Grobler, M.; Xiang, Y.; Zhu, Z. SPARSE: Privacy–Aware and Collusion Resistant Location Proof Generation and Verification. Global Communications Conference. IEEE, 2018, p. (To appear).
35. Conan, D.; Lim, L.; Taconet, C.; Chabridon, S.; Lecocq, C. A Multiscale Approach for a Distributed Event-Based Internet of Things. Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence & Computing. IEEE, 2017, pp. 844–852.
36. Sudhakar, T.; Inbarani, H.H. Spatial group mobility model scenarios formation in mobile ad hoc networks. International Conference on Computer Communication and Informatics (ICCCI). IEEE, 2017, pp. 1–5.