PairLoSh: Privacy Preserving Paired Location Sharing Crowdsensing Mechanism

Mehul Sen[1] and Dominic Adams[1]

[1]Golisano College of Computing and Information Sciences, Rochester Institute of Technology

***Abstract*—As the number of applications sharing data and the use of crowdsensing continue to grow, there is an increasing demand for data privacy. This encompasses all shared data, including location information. For companies relying on location data for services such as traffic estimation, ensuring user location privacy is challenging, particularly in the context of honest-but-curious servers. To address this issue, we propose a Paired Location Sharing mechanism called PairLoSh. Our system employs an agent to pair users together, who then exchange noisy data with each other. Subsequently, they aggregate their data and transmit it to the server via an encrypted channel. This approach ensures user privacy while sharing valuable location data. Additionally, we propose performance evaluation metrics to assess the effectiveness of our mechanism.**

I. INTRODUCTION

When multiple people or applications utilize or share the same data, it is crucial to ensure that the data remains safe and secure. Loss, theft, or corruption of the data can cause problems for everyone involved. Privacy-preserving methods can help protect shared data, preventing data leakage and safeguarding individual privacy.

Privacy preservation methods offer several benefits, including protecting sensitive data, maintaining data utility, ensuring regulatory compliance, and building trust. One key application of these methods is mobile crowdsensing, which presents several challenges, including incentivizing users to share their data.

Reverse-auction mechanisms have been employed to encourage data sharing in crowdsensing. However, a major issue is that the bid itself contains sensitive information. Although differential privacy solutions have been proposed to preserve bids and protect user privacy, these methods depend on trust in the platform collecting the data. In untrusted environments, this data remains vulnerable. Wang et al. suggested a solution involving agents as intermediaries between users and platforms [6]. This introduces a new type of actor - one agent per user - and requires users to place their trust in these agents.

Another challenge of mobile crowdsensing is the difficulty in preserving user locations. Location data is valuable to companies for various reasons, such as collecting traffic data or providing users with directions. In a mobile crowdsensing scenario, traffic data is the only aspect that falls under crowdsensing. The problem, however, is achieving privacy while maintaining location accuracy.

Our proposed solution addresses both issues by utilizing a single agent for all users and establishing a trust boundary between the user and all other parties. This method is designed to work with any type of data, but further research is needed to verify its effectiveness, which will be conducted as part of our project.

II. BACKGROUND

Mobile Crowdsensing: Mobile crowdsensing involves the collection and analysis of data from sensors on mobile devices by a large number of individuals. It relies on the concept of "crowdsourcing," which involves the distribution of tasks or data collection to a large number of individuals, usually through an online platform or mobile application. [6]

Trust Boundary: A Trust Boundary is the amount of trust an entity imparts to another entity. This includes permissions and access given to applications or users that might not be completely trusted. It is important to have a lower trust boundary because it helps to reduce the risk of security breaches and unauthorized access to sensitive data or resources.

Privacy-preserving Incentive Mechanisms (PPIM): PPIM is an incentive mechanism that ensures the privacy of user data in incentive-based systems or applications. It involves the use of privacy-preserving algorithms and protocols, through techniques such as data anonymization, data minimization, and data aggregation which ensure that user data is protected throughout the incentive process. [4]

Distributed Location Privacy Preserving Mechanisms (DLPPM): DLPPM is a set of techniques and protocols used to protect the location privacy of users in distributed networks. It addresses privacy concerns by allowing users to share their location information in a controlled and anonymous manner.

This is achieved through techniques such as data encryption, data aggregation, and obfuscation. [1]

Location Privacy using Gaussian Distribution: The Gaussian distribution, also referred to as the normal distribution or bell-shaped curve, is a continuous probability distribution characterized by a symmetric, bell-shaped curve. In this distribution, the probability of observing a particular value increases as it approaches the mean and decreases as it moves away from the mean. Researchers have applied this probability distribution to location data [5], offering a layer of privacy for location sharing.

III. Related Work

Xu et al. conducted research in the mobile crowdsensing field for their paper. They explored two novel techniques to address the issue of time-window dependent bids lacking clear privacy solutions. The proposed techniques are MST (Mechanism for Single Time window) and MMT (Mechanism for Multiple Time window), respectively. [7]

Jin et al. authored a paper on crowdsourced spectrum sensing, which aims to alleviate limited access to wireless networks. They proposed using reverse-auction-based mechanisms to incentivize users. Their paper demonstrates that the current framework is insufficient for privacy and proposes PriCSS as an alternative to reduce privacy loss. [2]

Both Jin et al. and Xu et al. highlighted that bids themselves contain sensitive information, leading to Wang et al.'s subsequent work in attempting to address this vulnerability. [2] [7]

Wang et al. introduced a novel privacy-preserving mechanism to protect users' true bids against an honest-but-curious platform. Their structure employs multiple agents to obfuscate users' true bids. A significant limitation of their mechanism is the expansion of the trust boundary to include these agents. [6]

Liu et al. examined privacy-preserving mobile crowdsensing (MCS) in dynamic scenarios using reinforcement learning. They proposed an approach that allows the platform to dynamically adapt its pricing policy to accommodate the varying privacy-preserving levels of participating users. Additionally, they built upon the concept of privacy-preserving incentive mechanisms. [4]

Dua, Singh, and Bapat discussed location privacy issues in the context of location-based services (LBSs). They emphasized the potential risks associated with continuous tracking of user locations and the generation of sensitive information, and expanded on the concept of location privacy-preserving mechanisms. [1]

Kim, Edemacu, and Jang conducted a survey of the current state of privacy preservation for location-based mobile crowdsensing. Their paper covers the present state of mobile crowdsensing, outlining various techniques and identifying pressing privacy issues. They presented three models: Mobile Crowdsensing with a Trusted Party (MCS-TP), where there is a server, workers, and a trusted third party responsible for data privatization; MCS-LOC for location-based data, in which the worker performs all privatization and only the worker and server are involved; and MCS-P2P, where workers share data with each other for privatization before sending it to the server. [3]

IV. Problem

Each of the different systems mentioned above has its own set of issues, with the two most significant being the trust boundary location and implementation costs. For the most secure systems, the trust boundary should be between an individual worker and all other parties to minimize the potential for malicious actors. The implementation costs or overhead mostly stem from Wang et al.'s work [6], as they introduced a new agent for each user, resulting in significant expenses.

V. Design Goals

To solve some of these problems, we propose a new mechanism called PairLoSh, which establishes a trust boundary at the user level while enabling users to share their obfuscated location data with the server. The mechanism is designed with the following goals in mind:

Privacy Preservation: The primary goal of PairLoSh is to protect users' location privacy by preventing adversaries from pinpointing any individual's true location. This includes safeguarding users' location data from other users within the system, honest-but-curious servers, and external adversaries.

Scalability: PairLoSh is designed to accommodate a large number of users, making it suitable for crowdsensing applications. The system should be capable of handling data sharing requests and user pairing allocations for multiple users concurrently.

Minimal Encrypted Communication: To enhance privacy and security, PairLoSh is designed to limit interactions between the server and other components, such as the agent and the users. It aims to minimize interactions between the agent and the server, and ensures that any direct interactions with users are encrypted.

Trust Boundary: PairLoSh seeks to position the trust boundary between the individual user and all other elements of the system. This approach reduces reliance on trust in other systems, making the mechanism more secure.

Minimal User Interactions: Finally, PairLoSh aims to minimize the interactions users need to make when sharing their location data with the server. This

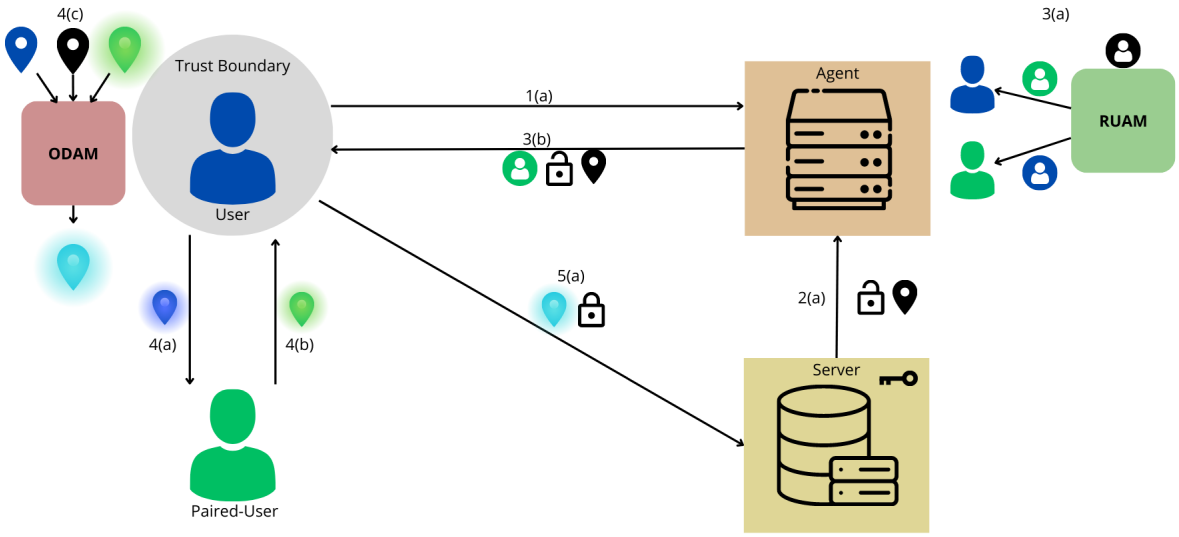

Figure 1. Architecture of PairLoSh

ensures that even devices with limited computational capabilities can implement this mechanism.

VI. Method

PairLoSh obfuscates a user's true location before sharing it with the server. The mechanism involves pairing, obfuscation, aggregation, encryption, and sharing for each individual.

Figure 1 displays the architecture of PairLoSh, and all the interactions within the mechanism for one User. This process would be repeated for all the users present in the system. A step-by-step breakdown of the mechanism is available in the following subsection. This mechanism employs two modules: the Obfuscated Data Aggregation Module (ODAM) on the User side and the Randomized User Allocation Module (RUAM) on the Agent side. These modules are further elaborated upon later in this paper.

This system employs an Agent to handle data sharing requests, allocate, and assign user pairings. A Server with limited interactions with the Agent is also used, sharing only its public key and the Server-assigned location. Furthermore, the interaction between the User and the Server, including the user's obfuscated location data, is encrypted using the Server's public key.

A. PairLoSh Interactions

1. **User-Agent Interactions**

(a) The User informs the Agent of their intention to share location data with the Server.

2. **Server-Agent Interactions**

(a) The Server transmits its public key and advertised location to the Agent. This not only ensures that the locations received from the client are encrypted but also provides Users with a means to estimate their perturbed location and allows the Agent to select user-pairings within a specific geographical area if necessary.

3. **Agent-User Interactions**

(a) The Agent employs the Random User Allocation Module to assign a user pairing for the User. These pairings are then shared with the involved users in a manner that prevents the Agent from disclosing the pairings to the Server.

(b) The Agent sends the user pairing, public key, and location to the User.

4. **User-Paired User Interactions**

(a) The User obfuscates its location using Gaussian distribution, connects with its Paired User, and shares the obfuscated location.

(b) The User receives the Paired User's obfuscated location.

(c) The User utilizes the Obfuscated Data Aggregation Module to generate the corresponding shared obfuscated location.

5. **User-Server Interactions**

(a) The User encrypts the shared obfuscated location obtained from the ODOM using the public key acquired in step 3(b) and sends it to the Server.

Afterward, the server can decrypt the perturbed location using its private key and acquire the shared obfuscated location of the user. This location is perturbed not only through the Gaussian distribution but also shifted in a direction based on the randomly determined user-pairing. Additionally, the User can set the level of shift for this location unless the Server enforces it.

B. Obfuscated Data Aggregation Module (ODAM)

The Obfuscated Data Aggregation Module (ODAM) is a User-sided module that generates

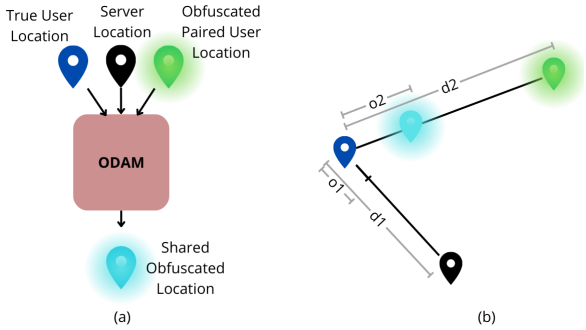

Figure 2. Obfuscated Data Aggregation Module

the shared obfuscated location of the user. Figure 2 illustrates the key idea behind ODAM. Figure 2(a) shows the inputs and outputs of the module. It takes in the User's location, the Server-provided location, and the Paired User's obfuscated location, and generates the shared obfuscated location. Figure 2(b) displays the workings of the module. In the figure, d_1 represents the distance between the User location and the Server location, while d_2 represents the distance between the User's location and the obfuscated Paired User's location. o_1 is a custom value that the user can set to increase or decrease the accuracy of their shared location. This value can range from 1% to 50% of the distance d_1. The module calculates the shared location based on o_2, which is determined as follows:

$$\frac{o_1}{d_1} = \frac{o_2}{d_2}$$

$$\frac{o_1}{d_1} * d_2 = o_2$$

The User's location is then obfuscated by o_2 to generate the shared obfuscated location of the User.

C. Randomized User Allocation Module

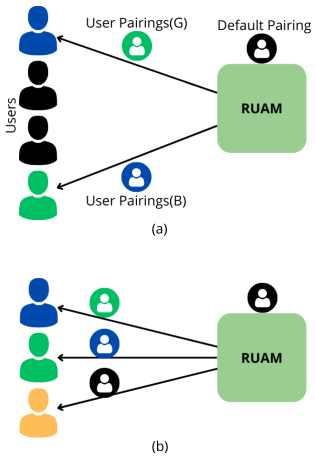

Figure 3. Randomized User Allocation Module

The Randomized User Allocation Module (RUAM) is an Agent-sided module used to assign user pairings. Figure 3 presents the key ideas behind RUAM. As shown in Figure 3(a), the module randomly selects two users who have advertised their intent to share their location data and pairs them together. It then generates the user pairing for each user and sends it to the corresponding users. Additionally, Figure 3(b) depicts a scenario where there are an odd number of users within the system; in this case, it uses a default location set during initialization as a user pairing with the odd-numbered user.

VII. Analysis

One of the primary features of this mechanism is the placement of the trust boundary. In other systems involving peer-to-peer communication, the trust boundary typically lies between the pair and everything else. In systems with an agent, it is situated between the agent and the server. However, in PairLoSh, the trust boundary exists between the individual and all other elements. This proves highly beneficial since it requires less trust in other systems, making it more secure if any part of the process becomes corrupted or if a bad actor is involved.

Regarding bad actors, there are three potential types in this mechanism. The first and most apparent type is a worker acting maliciously. Such a worker could be paired with another worker, potentially feeding them false information and sending inaccurate data to the server. However, they cannot reliably determine the exact location of the trustworthy worker they are paired with. Furthermore, unless there is an entire bot network involved, a single bad actor submitting false information will have minimal impact on the results, provided there is a sufficiently large sample size.

The second type of bad actor is the pairing agent. Their only possible course of action to tamper with the system would be to pair users in a manner that introduces noise into the system by connecting those who are farthest apart. This is unlikely, as users never share their locations with the agent.

The final potential bad actor is the server, even if it is merely honest-but-curious. The server cannot gather any personal information about the workers, as their data is obfuscated using others' data. Consequently, an honest-but-curious server has no impact on workers' privacy, as they are protected under this scheme.

One way the server could attempt to determine an individual's exact location is by repeatedly querying for their noisy location. Over time, the data would form a Gaussian distribution, with the person's true location at the central point. This is known as failing the Gaussian Distribution location privacy. Many mechanisms fail in this regard because simply adding

basic noise is not enough. However, our mechanism succeeds, as each time the server requests locations, it collects everyone's data, and each individual is paired with a new user. The server cannot determine which user the agent was paired with, so their data will appear different each time and never converge on a specific point, unless they somehow pair with themselves.

We chose location-based data for this system to facilitate ease of understanding and straightforward calculations. Aggregating data in a meaningful way was easily achievable with this type of data. However, this does not imply that the system is limited to location-based data only. ODOM can be readily adapted to work with various types of data; all that is required is a new scheme. This new scheme must be capable of aggregating data in a manner that remains useful to the server without introducing excessive noise that would render the data unreliable.

VIII. Evaluation and Future Work

Performance Evaluation Due to time constraints, this system has not been evaluated to check its performace and real-world applicability. Additionally, there are also several directions in which this mechanism can be further expanded and improved upon. To evaluate this mechanism, the following metrics can be used: Privacy leakage, Accuracy to distance ratio, and Deviation from true location.

- Privacy leakage: This would be the amount of disclosure of a User's location data to others within the mechanism. This would include, the paired-user, other users within the system, agent, server and a external party.
- Accuracy to distance ration: This would be the accuracy of the perturbed location given the distance between the user and the paired user.
- Deviation from true location: This would be the amount of deviation that a user's perturbed location has as compared to their true location.

Using these metrics, we can evaluate the performance of our mechanism using a location sharing crowdsensing experiment. This experiment could involve real-world location GPS dataset such as mobility traces of taxi cabs, or population of students within certain areas of the campus.

Future Work Future work could include the practical implementation and deployment of our mechanism, it could also involve incorporating Wang et al's Bid system [6] in order to incentivize sharing location data. Additionally, the modules used within this mechanism could be modified to work on data values other than location-data. Lastly, additional work can be done to reduce the complexity of our mechanism and make it computationally inexpensive for remote crowdseensing participants.

IX. Conclusion

In this paper, we introduce PairLoSh, a novel mechanism for sharing location data with a server while safeguarding the user's true location. PairLoSh utilizes two modules: a user-sided Obfuscation Data Aggregation Module and an Agent-sided Random User Allocation Module. This approach expands on prior research in mobile crowdsensing and integrates existing location privacy solutions, such as Gaussian distribution, while confining the trust boundary to the user.

Furthermore, we provide an analysis of the mechanism and discuss the potential impact of bad actors on its functionality. We also propose potential performance evaluation methods for our mechanism, assessing privacy leakage, accuracy-to-distance ratio, and deviation from the true location.

References

[1] Ankush Dua, Prasoon Singh, and Jyotsna Bapat. "Location Privacy-Preserving Mechanism - A Data-Driven Approach". In: *IEEE* (2021).

[2] Xiaocong Jin and Yanchao Zhang. "Privacy-Preserving Crowdsourced Spectrum Sensing". In: *IEEE* (2018).

[3] Jong Wook Kim, Kennedy Edemacu, and Beakcheol Jang. "Privacy-preserving mechanisms for location privacy in mobile crowdsensing: A survey". In: *Journal of Network and Computer Applications* (2022).

[4] Yang Liu et al. "An Incentive Mechanism for Privacy-Preserving Crowdsensing via Deep Reinforcement Learning". In: *IEEE Internet of Things Journal* (2020).

[5] Kato Mivule. "Utilizing Noise Addition for Data Privacy, an Overview". In: *ArXiv* abs/1309.3958 (2013).

[6] Zhibo Wang et al. "Towards Privacy-preserving Incentive for Mobile Crowdsensing Under An Untrusted Platform". In: *IEEE* (2019).

[7] Jia Xu, Jinxin Xiang, and Dejun Yang. "Incentive mechanisms for time window dependent tasks in mobile crowdsensing". In: *IEEE* (2015).