Encrypted Data Aggregation in Mobile CrowdSensing based on Differential Privacy

Michele Girolami
Information Science and Technologies Institute
National Research Council, ISTI-CNR
Pisa, Italy
michele.girolami@isti.cnr.it

Emanuele Urselli
Dep. of Computer Science
University of Pisa
Pisa, Italy
e.urselli@studenti.unipi.it

Stefano Chessa
Dep. of Computer Science and ISTI-CNR
University of Pisa and CNR
Pisa, Italy
stefano.chessa@unipi.it

***Abstract*—The increasing sensing capabilities of mobile devices enable the collection of sensing-based data sets, by exploiting the active participation of the crowd. Often, it is not required to disclose the identity of the owners of the data, as the sensing information are analyzed only on an aggregated form. In this work we propose a privacy-preserving schema based on differential privacy which offers data integrity and fault tolerance properties. In our schema, data providers firstly add a noise component to the sensed data and, secondly, they encrypt and send the cryptogram to the aggregator. The data aggregator is in charge of only decrypting the cryptograms, by preserving the identify of the data owners. We extend such schema by enabling data providers to submit multiple cryptograms in a time window, by using time-varying encryption keys. We evaluate the impact of the noise component to the generated cryptograms so that to evaluate the data loss during the encryption process.**

***Index Terms*—Differential privacy, Aggregation, CrowdSensing**

I. Introduction

The Mobile CrowdSensing (MCS) [1], [2] paradigm is designed to build representative data sets by exploiting the sensing capabilities of mobile devices that we daily use. More specifically, the idea is to exploit smartphones, smart watches or sensorized devices to collect data from the crowd, with a participatory approach. Traditionally, a MCS data collection campaign involves three actors: a set of volunteers' end-users providing sensed data, the mobile apps which asynchronously receive *tasks* and a MCS back-end [3]. A task defines an action to perform with or without the explicit intervention of the end-users. As for example, a task might require to use the microphone to sample the noise pollution in a geo-fenced area or to monitor the wireless signal coverage by exploiting the wireless network interface available with a smartphone. Often, it is not required to reveal the identify of the end-users, rather the MCS back-end only aggregates the collected data. In this work, we study a privacy-preserving mechanism for a MCS scenario guarantying anonymity of the volunteers' end-users, data integrity and fault tolerance of the collected data [4].

More specifically, we extend the approach followed in [5] which relies on the data aggregator. The role of the aggregator is to combine the data collected, such that it can read the composition of the data, but it cannot disclose the identify of the data owners. In particular, data providers collect sensing data, they add a noise component and finally then they cipher the resulting information. The aggregator receives and combines such cryptograms, it can decode the aggregated version but it cannot read who produced the cryptogram. We extend the schema described in [5] by introducing the temporal dimension. Indeed, in a MCS scenario data providers can upload data asynchronously, this is the case of a sensing task which requires to sample a specific environmental parameter for 24h. In this example, data providers can upload data at periodical intervals or they can upload data in burst. In our schema, data providers upload cryptograms whose key varies according to the time, so that each cryptogram varies as time progresses. Moreover, data providers can transmit an arbitrary number of cryptograms during a time window, so that to match with the sampling frequency of sensors available with smartphones. We evaluate the impact of the cryptogram's noise component to measure how much information the aggregator can re-build after the decoding phase. To this purpose, we implement a simulator designed to test the encoding/decoding steps executed by data providers and the aggregator by considering an environmental monitoring task. Our experiments reproduce the behavior of a number of data providers collecting a varying number of temperature readings extracted from Weather Underground service for a time period of 2 hours. Our results show that the, according to the number of generated cryptograms, the aggregator can re-build a median of the temperature with a very reduced error, always below 0.02 Celsius degrees.

II. Background and Related work

Several techniques have been proposed to anonymize the identify of the MCS volunteers, as reported in [6], [7]. Authors of [8] propose a solution based on differential-privacy which obtains low aggregation error but at a high communication costs to transfer the ciphered information. A different approach is adopted in [9], [10], in which authors explore a solution based on data aggregation at the level of time-series with a lightweight encryption. Authors of [5] propose an encryption schema specifically tailored on a MCS scenario. In the reminder of this section, we briefly summarize such solution, as it represents the approach that we extended in this work. Authors of [5] identify 3 architectural components: the

devices, an aggregator and the Cloud. In particular, authors address the following requirements:

- Efficient group: a strategy to partition the MCS users in different groups. Users' devices can join/leave the group they belong to, and they can refresh the adopted encryption mechanism;
- Data integrity: the aggregator can verify the integrity of the collected data so that to prevent a man-in-the middle attack;
- Fault tolerance: a strategy to prevent data corruption during communications between end-devices and the aggregator.

The considered scenario is characterized by a set of devices $u_i, i \in [1, n]$ collecting sensing information at different time periods. We refer to $x_{i,t}$ as the data collected by u_i at time t. More specifically, authors detail a periodic aggregation schema, through which devices can periodically encrypt and transmit sensed information to the aggregator. The cryptogram sent by u_i at time t is composed by:

$$c_{i,t} = x_{i,t} + \cup R_{i,t} \tag{1}$$

where $x_{i,t}$ represents the collected data summed with a noise given by the difference between two Gamma distributed random variables of shape α: $\hat{G}_i(\alpha, \lambda) = G_{i,1}(\alpha, \lambda)G_{i,2}(\alpha, \lambda)$. As discussed in [5], the perturbation given by $\hat{G}_i(\alpha, \lambda)$ is not enough to obfuscate $x_{i,t}$ therefore, u_s adds the factor $\hat{R}_{i,t}$ which is obtained from the the cipher keys $R(i, i-1)$ and $R(i, i+1)$ used at time t by device i to encrypt $x_{i,t}$. $\hat{R}_{i,t}$ is given by:

$$\hat{R}_{i,t} = H_2(F_{H_1(R(i,i+1))}(t)) - H_2(F_{H_1(R(i,i-1))}(t)) + s_i \mod q \tag{2}$$

$\hat{R}$ is obtained by 2 one-way hash functions H_1, H_2 and F is defined as a PseudoRandom function using H_1 as input parameter[1]. Every device can apply Equation 1 to build the cryptogram and to send it to the aggregator with a wireless link, e.g. WiFi or Bluetooth. In turn, at every time step, the aggregator receives a sequence of cryptograms that it aggregates (sums) as follows:

$$\sum_{i=1}^{n} c_{i,t} = \sum_{i=1}^{n} x_{i,t} + \sum_{i=1}^{n} \hat{R}_{i,t} \tag{3}$$

It is worth to notice that the the sum of all perturbations $\hat{G}_i(n, \lambda), \forall u_i$, returns a Laplacian random variable (r.v) of mean $\mu = 0$ and scale λ: $Lap(\mu, \lambda) = \frac{1}{2\lambda} \exp(-\frac{|x|}{\lambda})$, commonly used with differential privacy problems.

III. System Model

We consider a Mobile CrowdSensing (MCS) platform that involves a number of end-users. Each end-user installs in her/his smartphone a mobile application interacting with the MCS platform to implement sensing tasks, that may or may not require the assistance of the end-user. The platform is managed by a centralized component, (hereafter MCS manager) that injects tasks and that collects the results. Each task specifies the sensing activity to be executed by the mobile app, and it may be directed to a specific subset of apps, depending on the profile of their end-users.

We assume that a task, in general, produces a stream of sensed data flowing from the apps to the MCS manager. The MCS manager, in turn, stores the collected data for further analysis. To the purpose of making more efficient the process of data collection, especially when a task involves a large number of apps, the MCS platform relies on an intermediate layer of edge computing servers, each covering a given logical or physical area. Such layer is in charge of collecting and aggregating the data streams produced by the apps operating in their respective area of coverage. We refer to such layer as the aggregator.

More specifically, in this work we focus on tasks for which it is possible to perform data aggregation. As a matter of example, let us consider a task to measure the quality of a wireless signal, e.g. WiFi or LTE propagated by a set of base stations. The task activates the app of the end-user to periodically measure the received signal strength indicator (RSSI) estimated by the the wireless interface. For each data read from the wireless interface, the app tags it with the current timestamp and sends it to the local aggregator. In turn, the aggregator sums all the received data with a matching timestamp. Finally, the aggregator averages the sensed information and sends it to the MCS manager.

In the rest of the paper, we assume that the MCS platform involves n end-users, each of which has installed in her/his smartphone an instance u_i ($i \in [1, n]$) of the MCS app. We also denote by M the MCS manager, and by a_j ($j \in [1, p]$) the aggregation servers previously introduced. In a given period of time, each a_j aggregates the data received from u_i. A task T sent by M to the MCS apps, is defined for a time frame $[t, t']$ and it requests the apps to perform a periodic sampling with period s. We assume that the interval $[t, t']$ is slotted, where each slot lasts a period $l \geq s$, and that each aggregator performs elaborated the data within a single slot. Focusing to a given area covered by the aggregator a_j, we assume that all the apps in A_j are synchronized with the beginning of each slot, hence each app produces a number of sampled data (each tagged with the corresponding timestamp) in each slot, and all these data are aggregated together. Note that the messages exchanged between the apps and the aggregator may be delayed or even lost, and the apps themselves are not necessarily reliable as they can be disabled by their end-users without any notice or a end-user herself may migrate (along with his smartphone and MCS app) to the area covered by another aggregation server. Hence the aggregation process should be tolerant to missing data.

IV. Privacy and encrypted data aggregation in Mobile CrowdSensing

In the described model the data the aggregator might be not a trusted agent in our MCS architecture. If this is the case,

[1]Authors adopt HMAC-SHA256 as PRF function

then the aggregator may make improper use of the data and possibly attempt attacks to infer other information about the subscribers. It should be observed however that the same issues remains even if the MCS platform does not use intermediate aggregation servers. In this case, in fact, the subscribers data is available in cleartext to the MCS manager that may also make improper use of the subscribers data. Note in fact that, to the specific purpose of the MCS campaign, the objective is to obtain an aggregation of the data and not the single individual data alone. For this reason, even using an encrypted transmission of the data from the apps to the MCS manager would not provide any privacy guarantee to the subscribers under this respect. Hence, we consider an approach in which the data are aggregated in an encrypted form, so that the aggregation server, which does not have the decryption keys, does not have access to the clear-text data, and the MCS manager receives only reports of aggregated data that hence do not bring any individual information about the subscribers. More specifically we adopt the encryption method proposed in [5] and described in Section II. The proposed schema, is sensible to the loss of data in the communication from the apps u_i to the aggregator, which may happen for several reasons. As for example, the data are delayed, the user is offline or the wireless links between users and the aggregator is subject to interference. To cope with these issues, authors of [5] propose the use of a buffering mechanism of the future messages (that can be used if the aggregation does not operate in strict real-time), and that lets the aggregator to fill the missing data in order to compute correctly the aggregation. Let us consider the case in which the n app instances are instructed to collect data at a given frequency during the time interval $[t, t']$, and that these data need to be aggregated at each time slot, so that in each slot the aggregator expects to receive f samples from each u_i. In practice, referring to a given slot, say $[t_1, t_2]$, the aggregator receives from each u_i a number of samples equal to $f_i \leq f$. In order to perform a correct aggregation, the aggregator needs to compute the average by considering the same number of samples from each u_i. As in general this is not possible, the aggregator computes $f_M = max_{i\in[1,n]}\{f_i\}$, and it considers the computation only the data from the u_i such that $f_i \geq \delta \times f_M$, where $\delta \in [0, 1]$ is a system parameter. In order to replace the missing data with potentially meaningful data for a given u_i, the aggregator includes in the computation also the first $f_M - f_i$ data provided by the buffering mechanism of the aggregator of the data provided by u_i [5]. However, the problem in this approach is that, as these data are replacement and hence not real data taken in the slot, this may alter the result of the aggregation. This is particularly problematic in a MCS context where the amount of loss data may be significant. A second aspect to be considered is the effect of the random noise inserted by the u_i before the encryption (see equation 1 to ensure the differential privacy property, which may also alter the final result of the data aggregation.

V. Experimental Settings and Results

To the purpose of assessing the effect of noise and loss of data in a MCS platform, we developed a simulator modeling the interaction between an aggregator and n applications. To feed the simulation with real data, we considered environmental data (specifically temperature readings from Bologna city, Italy) measured by 10 meteorological stations available by Weather Underground[2]. The stations sample the temperature every 10 minutes over a period of 2 hours, we consider data collected on November 4th, 2016. The encryption algorithm is configured by setting the parameters of the Gamma distribution to $\alpha = 1/10$ and $\lambda = 2$. The simulator makes an exhaustive analysis of all possible combinations of data received by the stations, and it compares the outcome of the aggregation against an aggregation conducted in the same conditions but without encrypted aggregation.

The results of the simulation are summarized in Figures 1 to 3. Figure 1 reports the real and the computed average, by varying the number of noisy input values that are received by the aggregator (this simulates the presence of missing data from some of the sources), with the respective 95% confidence interval (the shaded area). The aggregation is performed with 10 readings, reproducing a scenario with low data collected from end-users. Figures 2 and 3 report the results with a higher number of readings, 80 and 120 readings, respectively. These scenarios reproduce a more significant scenarios in which the intensity of noise injected at the encryption time varies between 10% to 90%. As a general consideration, the reported figures show that the system is tolerant to noise since the error due to the encrypted aggregation is affected by an error that is limited to around 0.02 Celsius degrees in all cases.

Finally, Figure 4 shows how the average temperature value vary with a varying number of readings, from 1 to 120. The box plots show an increasing trend of the noisy of the resulting average. Specifically, each box plot shows the median of the average temperature for a given number of readings, and the 25-*th* and 75-*th* percentile, as well as the outliers. The increase of size of the box plots depends from the fact that each data brings to the aggregator its additional noise, and this suggests that the aggregation should be execute on limited time frames in order to avoid the divergence between real data (sampled buy the stations) and the results of the aggregator.

VI. Conclusions

We explore in this work how the Mobile CrowdSensing paradigm can be enriched with a privacy-preserving encryption schema, able to guarantee the privacy of the end-users, e.g. the data providers, data integrity and fault tolerance. We extend the work presented in [5], by introducing a temporal management for the cryptograms generated by the data providers. This extension allows to generate an arbitrary number of cryptograms which, in turn, can be aggregated and sent to the MCS back-end. We analyze the effect of the noise component added to the cryptograms with the goal of measuring how

[2]www.wunderground.com

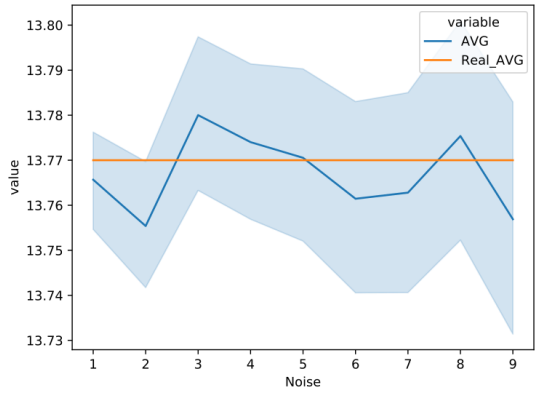

Fig. 1. Average temperature obtained with 10 sample readings.

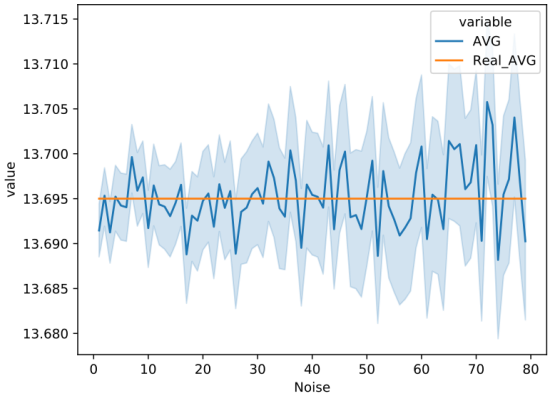

Fig. 2. Average temperature obtained with 80 sample readings.

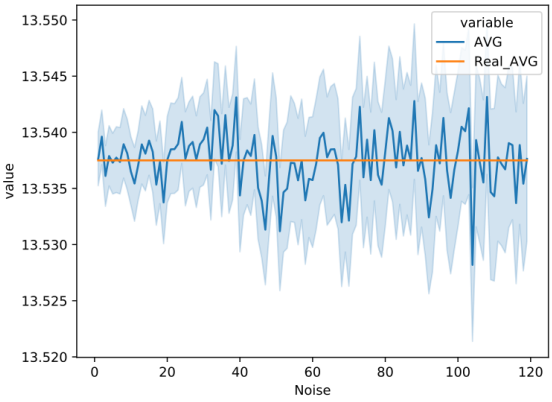

Fig. 3. Average temperature obtained with 120 sample readings.

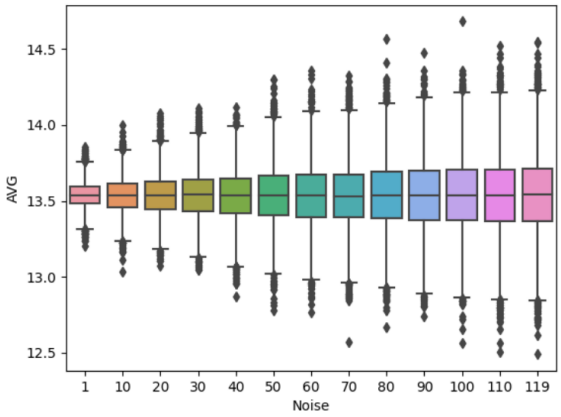

Fig. 4. Box plot showing the contribution of the 10 data providers with 120 readings to re-build the average temperature.

much information is lost during the aggregation level. To this purpose, we simulate the generation of ciphered temperature readings extracted from Weather Underground, by measuring the difference between the actual and aggregated median value of the temperature for as period of 2 hours. Our results show that such difference is always below 0.02 Celsius degrees. Further studies are however necessary to extend the time frame of the aggregated data, and to analyze other kind of data that possibly has a larger variability than the temperature.

References

[1] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2419–2465, 2019.

[2] J. Liu, H. Shen, H. S. Narman, W. Chung, and Z. Lin, "A survey of mobile crowdsensing techniques: A critical component for the internet of things," *ACM Trans. Cyber-Phys. Syst.*, vol. 2, no. 3, jun 2018. [Online]. Available: https://doi.org/10.1145/3185504

[3] V. S. Dasari, B. Kantarci, M. Pouryazdan, L. Foschini, and M. Girolami, "Game theory in mobile crowdsensing: A comprehensive survey," *Sensors*, vol. 20, no. 7, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/7/2055

[4] F. Khan, A. Ur Rehman, J. Zheng, M. A. Jan, and M. Alam, "Mobile crowdsensing: A survey on privacy-preservation, task management, assignment models, and incentives mechanisms," *Future Generation Computer Systems*, vol. 100, pp. 456–472, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18315632

[5] J. Chen, H. Ma, and D. Zhao, "Private data aggregation with integrity assurance and fault tolerance for mobile crowd-sensing," *Wireless Networks*, vol. 23, no. 1, pp. 131–144, Jan 2017. [Online]. Available: https://doi.org/10.1007/s11276-015-1120-z

[6] L. Wang, D. Zhang, D. Yang, B. Y. Lim, and X. Ma, "Differential location privacy for sparse mobile crowdsensing," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 2016, pp. 1257–1262.

[7] J. Chen, H. Ma, D. Zhao, and L. Liu, "Correlated differential privacy protection for mobile crowdsensing," *IEEE Transactions on Big Data*, vol. 7, no. 4, pp. 784–795, 2021.

[8] J. Won, C. Y. T. Ma, D. K. Y. Yau, and N. S. V. Rao, "Proactive fault-tolerant aggregation protocol for privacy-assured smart metering," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 2804–2812.

[9] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 735–746. [Online]. Available: https://doi.org/10.1145/1807167.1807247

[10] K. Emura, H. Kimura, T. Ohigashi, T. Suzuki, and L. Chen, "Privacy-preserving aggregation of time-series data with public verifiability from simple assumptions and its implementations," *The Computer Journal*, vol. 62, no. 4, pp. 614–630, 2019.